Learning diary and answers

These are optional fields about your identity, leave empty if you want to remain anonymous:

Student name:

Email address (prefer your @students.oamk.fi email if you have such address):

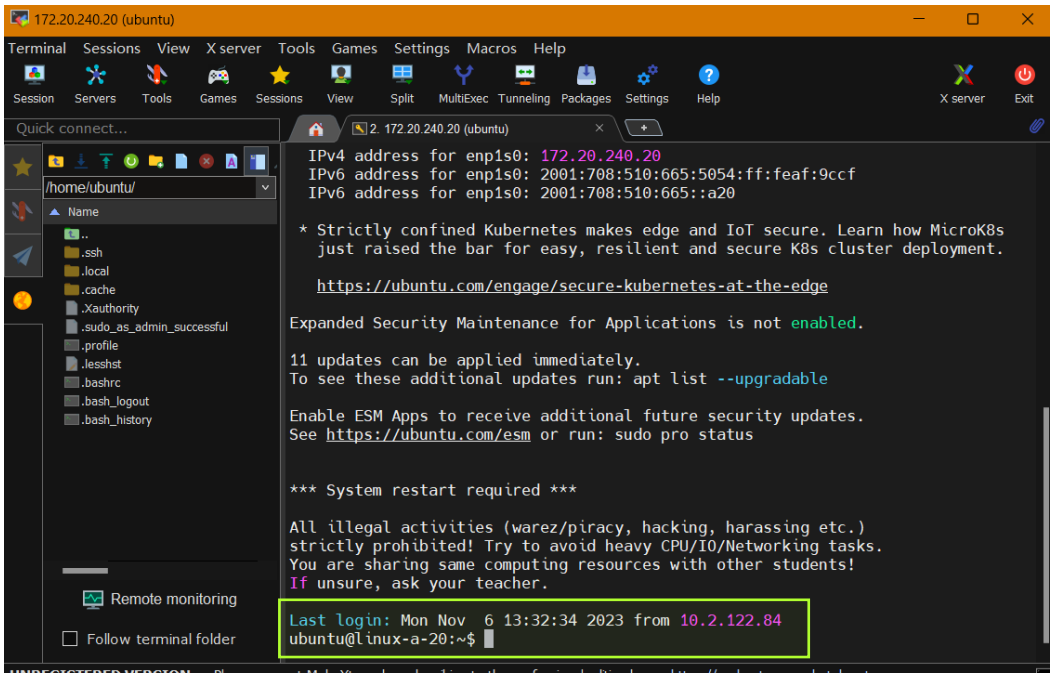
Save the final version of this document as PDF and submit it for peer reviews via Moodle's workshop tool before the deadline. Last course week is for peer reviews.

Some courses may have 5 weeks, and some may have 8 weeks of assignments. This is a generic learning diary template. Adapt and edit the document accordingly.

Week 1

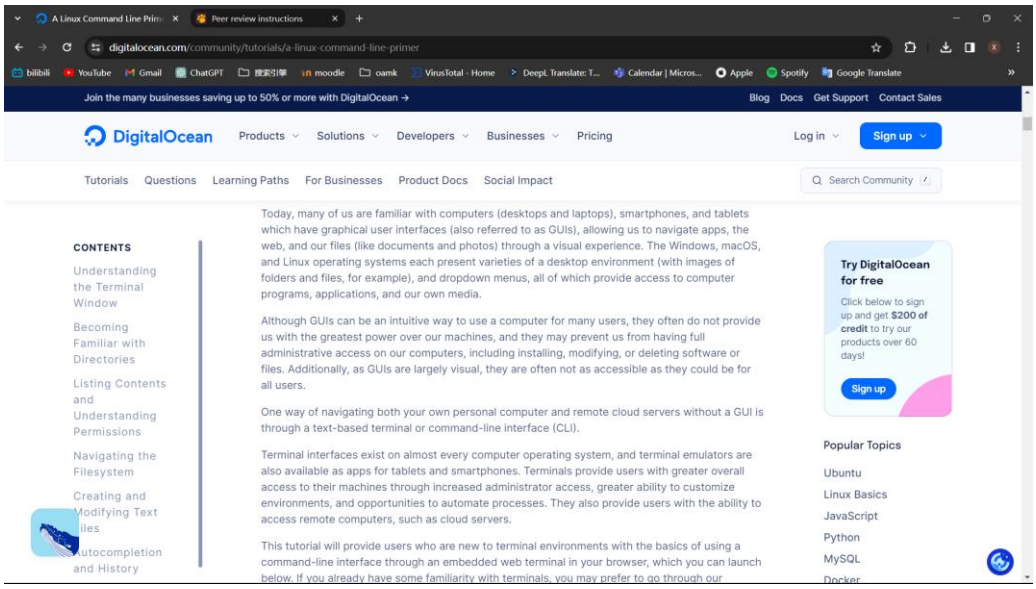
Question 1: Install Ubuntu Linux (or whatever Linux distribution you prefer) to VMware, VirtualBox, Hyper-V or KVM (or use Windows 10/11 WSL/WSL2). You can skip this if you are already using personal Linux desktop(s) or server from Oamk.
Note: Virtualbox has often compatibility issues with WSL2

Answer 1:



Question 2: Browse this Linux Command Line Primer

Answer 2:



Question 3: Describe following commands and concepts:

Answer 3:

- **man, apropos**: Manual pages have brief descriptions. The 'apropos' command searches these descriptions for a specified 'keyword', which can be a regular expression (-r), contain wildcards (-w), or match an exact term (-e). To avoid misinterpretation by the shell, it might be necessary to enclose the 'keyword' in quotes or use a backslash () to escape special characters.
- **man date**: To showcase the current time in the provided FORMAT or adjust the system date, the command requires specific options. The mandatory arguments for long options are also obligatory for short options.
- **ls**: To display a list of files within a directory, you can use the ls command.
- **ls –-help**: The ls command lists information about the specified FILEs, defaulting to the current directory if none are specified. If no sorting options like -cftuvSUX or --sort are given, entries are sorted alphabetically.
- **date**: Display you the exact time of day month the number of week time time zone and year .

  For example :Thu Nov 2 11:00:18 EET 2023
- **date –-help**: Display the current time in the given FORMAT, or set the system date.
- **cd cd / cd ~ cd ..:** The 'cd' command stands for "change directory" and is used to navigate through the file system. To move from the current directory (/home/sammy) to the newly created directory (/home/sammy/files), you can use the command 'cd files'.

  Additionally, to move to the primary directory of the server from any location within the file system, you can use 'cd /'. This command takes you to the root directory (/), which is the top-level directory in the file hierarchy.

The 'cd ..' command is used to navigate to the parent directory of the current directory. For instance, if you are in a directory '/home/sammy/files', executing 'cd ..' would take you up one level to '/home/sammy'.

The 'cd ~' command is used to change the current directory to the user's home directory. It takes you back to your home directory regardless of your current location in the file system.

- **cd -**:When you execute cd -, it will take you back to the directory you were in before the most recent cd command.
- **cd ..** :Linux leverages dot notation to help users navigate via relative paths. A single . stands for the directory you are currently in, and a double .. stands for the parent directory. So, from where we currently are (/home/sammy/files), we can use two dots to return to the parent /home/sammy user directory, like so: Another important symbol to be familiar with is ~ which stands for the home directory of your machine. Here, our home directory is called /home/sammy for the sammy user, but on a local machine it may be your own name as in sammy-shark/.
- **ls -s aaaa***: The 'ls -s aaa*' command in Unix and Linux lists files in the current directory that match the pattern aaa* and displays them along with their respective sizes in kilobytes.
- **ls -lat**: The 'ls -lat' command in Unix and Linux is utilized to display files and directories in a directory, sorted by modification time in descending order, showing the newest files or directories first, followed by older ones.
- **pwd**: The pwd command stands for “present working directory,” and it lets you know where you are within the current filesystem.
- **chown**: The chown command is a Unix and Linux command used to change the ownership of files and directories. “Chown” stands for “change owner.” It allows you to specify a new owner and, optionally, a new group for a file or directory.For example:chown xxx lename.
- **chmod**: “Chmod” stands for “change mode.” It allows you to specify who can read, write, and execute a file or directory. For example:chomod xxx filename.
- **chgrp**: The 'chgrp' command in Unix or Linux stands for "change group." It permits the system administrator to modify the group ownership of a file or directory. This ownership designation influences access permissions for users based on their group membership, determining their level of access to the file or directory.
- **chmod 644 file**:change mod of file ,which allows Owner: read (r) and write (w)Group: read (r)Others: read (r)
- **chmod g+x myfile**: The command chmod g+x myfile adds execute (x) permission for the group (g) on the file named myfile. After executing this command, members of the group associated with the file will be able to execute the file if it is a script or a binary executable.
- **which**: When you run which followed by the name of a command, it tells you the location of the executable file associated with that command.
- **rm** : The 'rm' command in Unix or Linux is used to delete or remove individual files. To remove a specific file, you need to specify the file's name as an argument after the 'rm' command.

Learning diary and answers

- **rm -r mydoc**: Using the 'rm' command with the '-r' flag in Unix or Linux allows for the recursive removal of the 'students' directory and all its contents. The '-r' flag signifies "recursive" and by passing the folder name 'mydocthe argument, all files and subdirectories within 'mydocl be deleted.
- **cp primary secondary** : The command cp primary secondary does not have a specific meaning by itself in Unix or Unix-like operating systems. It appears to be a command with placeholders for "primary" and "secondary," which would typically represent the source and destination files or directories you want to copy.
- **mv file2 file1** : mv: This is the command for moving (or renaming) files and directories.

  In this case, file2 will be moved to the /path/to/destination/ directory and renamed to file1.
- **wc -l myfile**: wc - print newline, word, and byte counts for each file . -l, --lines print the newline counts
- **mkdir mydata**: The 'mkdir mydata' command in Unix or Linux creates a new directory named 'mydata'. The 'mkdir' command stands for "make directory," and when this command is executed, the name of the folder ('mydata' in this case) is passed as an argument. In command terminology, an argument represents the input provided to the command, similar to an object being acted upon by a verb in natural language grammar. In this scenario, 'mydata' is the argument acted upon by the 'mkdir' command to create the directory.
- **rmdir mydata** : The 'rmdir mydata' command in Unix or Linux is used to remove directories. The 'rmdir' command, short for "remove directory," is specifically designed to delete empty directories. If you attempt to delete a directory containing files or subdirectories, the command won't work, as 'rmdir' solely works on empty directories.
- **more, less**: more is a filter for paging through text one screenful at a time. This version is especially primitive. Users should realize that less(1) provides more(1) emulation plus extensive enhancements.
- **file**: file — determine file type
- **stat**: stat - display file or file system status
- **df**: df - report file system disk space usage
- **ln**: ln - make links between files
- **which**: which returns the pathnames of the files (or links) which would be executed in the current environment, had its arguments been given as commands in a strictly POSIX-conformant shell. It does this by searching the PATH for executable files matching the names of the arguments. It does not canonicalize path names.
- **Whereis**: locate the binary, source, and manual page files for a command
- **find**: find - search for files in a directory hierarchy
- **touch**:The 'touch' command in Unix or Linux is employed to create a new file or update the access and modification timestamps of an existing file. To utilize 'touch', you simply need to enter the command followed by the name of the text file you wish to create or modify. An example would be: 'touch filename.txt' where 'filename.txt' is the name of the text file you want to create or update.
- **touch mynewfile** : create a new file which name is mynewflie
- **cp /tmp/test.txt ~/temp/:**

- **cp** :In order to have a template for the other students, we can copy the sammy.txt file to create more files. To do this, we can use the cp command, which stands for copy. This command works similarly to the mv command, taking the original file as the first argument, and the new file as the second argument.

  The command provided (cp /tmp/test.txt ~/temp/) is used to copy a file named test.txt from the /tmp directory to a directory called temp inside your home directory (~ represents your home directory).

- **echo** :If we want to create a text file that is initialized with text, we can use the echo command, which is used to display strings of text in Linux.We can use echo directly on the command line to have the interface repeat after us. The traditional first program, "Hello, World!", can be written with echo like so:Named for Echo of Ovid's Metamorphosis, the echo command returns back what we request. Named for Echo of Ovid's Metamorphosis, the echo command returns back what we request. In this case, it echoed, "Hello, World!" On its own, however, the echo command does not allow us to store the value of our text into a text file. In order to do that, we will need to type the following: The above command uses echo, then the text we would like to add to our file in quotes, then the redirection operator >, and finally the name of our new text file, sammy.txt.

- **cat** : Short for concatenate, the cat command is very useful for working with files. Among its functions is showing the contents of a file. If we were to run cat on the empty file ocean.txt, we would receive nothing in return as there is no text in that file. We can add text to this existing file with echo as well. Let's add a quote from Zora Neale Hurston to the file.

  Try typing cat along with the first few letters of one of the text files you have been working on — for example, cat sa. Before you finish typing the whole file name of sammy.txt, press the TAB key instead. This should autocomplete the full file name, so that your terminal prompt displays the following:

  Another shortcut is to press the UP arrow key, which will let you cycle through the most recent commands you have run. On a new line with a blinking cursor, press the UP arrow key a few times to have quick access to your previous commands.

  If you need to replicate all the commands you have done in your terminal, you can also summon the entire history of this session with the aptly named history command:

- **Curl**: From here, we'll use the curl command to transfer data from the web to our personal interactive terminal on the browser. The command curl stands for client URL (web address).

- **Exit**: When you are done with a terminal session, and especially when you are working on a remote server, you can exit the terminal with the exit command. Once you feel comfortable with what you have achieved in this session (as you won't be able to restore it), you can type the following, followed by ENTER to leave the terminal.

Question 4: What is the difference between Linux kernel and GNU/Linux distribution?

Answer 4:
Linux, as a kernel derived from Unix, forms the fundamental core of an operating system (OS). An OS comprises two key components: the Kernel and the Shell. The Kernel operates in the background, translating user commands into machine-specific signals. It serves as the foundational core of the OS. On the other hand, the Shell acts as the intermediary between users and the Kernel. The Shell can manifest as either a command line interface (CLI) or a graphical user interface (GUI).

In the context of GNU/Linux distributions, it's essential to recognize the fusion of the Linux kernel and the associated Shell, culminating in a comprehensive package known as a distribution or "distro." This amalgamated system incorporates not only the kernel and the Shell but also includes supplementary software applications. Linux offers diverse Shell options, including bash, sh, dash, zsh, as well as graphical interfaces such as GNOME, KDE, Unity, Xfce, Pantheon, among various others.

Linux distributions exemplify this combination, with notable examples including Ubuntu, Debian, Fedora, Kali, RedHat, and ElementaryOS. These distributions package the Linux kernel, a selected shell, and an array of software applications tailored for diverse user needs.

Expanding this analysis to other ecosystems, Windows, developed by Microsoft, constitutes a distribution housing its own Windows kernel and graphical user interface (GUI). Similarly, OS X, crafted by Apple, rests on a modified Unix kernel and an Apple-designed GUI.

Moreover, Android, emerging from the Android Open Source Project, relies on the Linux kernel paired with a variety of available graphical user interfaces or user interfaces (UIs).

Question 5: Name some very common Linux distributions.

Answer 5: Linux distributions like Ubuntu, Debian, Fedora, Kali, RedHat, and Linux Mint cater to various user needs within the Linux ecosystem.

Ubuntu: Offers Desktop, Server, and Core editions, featuring a user-friendly interface and bundled free software like LibreOffice, Firefox, Thunderbird, and games such as Sudoku and chess. Recognized for enhanced security compared to Windows, it allows extensive customization for tailored user environments.

Debian: Known for stability, Debian allows experimenting with Unstable package versions while providing an extensive package repository. Its elongated upgrade cycle enables prolonged use of a single version, reducing the need for frequent software upgrades, making it suitable for servers.

Linux Mint: Gained popularity since its 2006 beta release for its user-friendliness, resembling the Windows interface, thereby easing the transition for new users. It comes preloaded with useful apps like Firefox, LibreOffice, and boasts an intuitive Software Manager for easy app installations.

Question 6: What is GPLv2/v3 license? And BSD style license?

Answer 6:
The GNU Public Licenses (GPL) versions GPL v2 and GPLv3, overseen by the Free Software Foundation (FSF) and authored by Richard Stallman, aim to promote free software usage. However, they differ in key aspects.

GPLv3, being newer and longer than GPLv2, emphasizes improvements like patent indemnity, internationalization, and remedies for license infringement. It clarifies distribution and derivative work definitions and allows code combinations with additional requirements, making it more compatible with various licenses compared to GPLv2.

Question 7: What is (operating system) shell?

Answer 7: The shell in an operating system acts as the user's interface to execute commands and perform tasks. It provides a command-line environment where users input instructions to access and control the OS services.

As the outermost layer, the shell incorporates a programming language, enabling users to manage files, control processes, and execute programs. It interprets user commands, executes them, and displays resulting outputs.

Users interact with the shell through a command prompt (often represented by a $ sign) after logging in. They input commands, and upon hitting Enter, the shell executes them, presenting outputs or redirecting them as needed.

The command line represents where users type commands, marked by the shell prompt, facilitating communication with the OS for executing immediate actions or scripting command sequences stored in files for complex operations.

Question 8: What are case sensitive file names?

Answer 8: Case sensitivity refers to how an operating system or file system treats uppercase and lowercase letters within file names or directory names. In a case-sensitive environment, uppercase and lowercase letters are considered distinct, meaning "CAT.txt and "cat.txt be recognized as two separate and distinct files. Conversely, in a case-insensitive system, these variations would be treated as the same file.

Question 9: Describe common purpose of files and directories in “/etc”, “/usr/bin” and “/var”.

Answer 9:

/etc:

Example: Configuration files for network settings (e.g., /etc/network/interfaces), system-wide application settings (e.g., /etc/apt/sources.list for package management), and user account configurations (e.g., /etc/passwd for user information).

/usr/bin:

Example: Executable binary files for essential commands like "ls" (list files), "cp" (copy files), "mv" (move files), "grep" (search text), and "mkdir" (create directories). These commands are executable through the command-line interface.

/var:

Example: Storage of dynamic data that frequently changes during system operation, including log files like /var/log/syslog, spool directories (/var/spool/mail for mail queues), temporary files (/var/tmp), and databases that record system changes. For instance, web server logs (e.g., /var/log/apache2/access.log) and email spools are kept here.

Question 10: What is shell PATH? What is the difference between absolute and relative path?

Answer 10: The shell PATH is an environment variable containing directories where the shell searches for executable files when a command is entered. For instance, typing "ls" triggers the shell to look for an executable file named "ls" in the directories listed in the PATH variable.

Absolute Path: An absolute path denotes the complete file or directory location starting from the root directory. For example, "/home/user/documents/file.txt" indicates the file "file.txt" resides in "documents" within "user," located in "home," and so on.

Relative Path: In contrast, a relative path specifies a file or directory location relative to the current working directory. For instance, being in "/home/user/documents," a relative path to "file.txt" in the same directory is simply "file.txt."

Question 11: What is the purpose of tilde character (~) for most Linux shells. For example ls ~/.

Answer 11: When used in a path, the tilde represents the user's home directory, simplifying navigation.

On the other hand, the period (.) in a filename doesn't inherently convey any specific meaning. However, certain programs like "ls" typically hide files whose names begin with a dot unless specifically instructed to display them. These files are often considered "hidden" due to their filenames starting with a dot (.).

For example, if a file is named ".config" in a directory, when using the "ls" command without additional options, it might not display this file by default, considering it as hidden. Explicitly specifying options with "ls," such as "ls -a," will reveal these "hidden" files, showing filenames starting with a dot.

Question 12: How do you recognise a hidden file in any common Unix/Linux file systems?

Answer 12: In Unix and Linux file systems, hidden files and directories are indicated by a leading dot (.) in their names. For instance, a hidden file could be named ".example.txt," while a hidden directory might appear as ".config."

By default, commands like "ls" don't display these hidden files when listing a directory's contents. To view hidden files, the "-a" or "--all" option can be used with the "ls" command.

ls -a This command will display all files and directories, including the hidden ones, in the current directory.

Question 13: What is the meaning of “piping data between commands”?

Answer 13: "Piping data between commands" in Unix-like operating systems refers to the process of taking the output of one command and using it as the input for another command. This is achieved using the pipe symbol "|".For example, if you have two commands, A and B, you can use a pipe to send the output of command A directly to command B, without saving it to a file or using temporary storage. The syntax looks like this:

commandA | commandB

Here, the output of commandA is fed as input to commandB. This allows for a seamless flow of data between different commands, enabling them to work together to perform more complex tasks.

Question 14: What are seti-uid (suid) and set-gid (sgid) bits for file permissions?

Answer 14: Setuid (Set User ID):

This permission bit, when set on an executable file, allows the program to run with the privileges of the file's owner. For example, if a file is owned by the root user and has the setuid bit set, when a regular user executes that file, the program will temporarily gain the root user's privileges for its execution.

Setgid (Set Group ID):

When applied to an executable file, the setgid permission causes the program to run with the privileges of the group associated with the file. For instance, if a file belongs to a specific group and has the setgid bit set, any user executing that file will run it with the permissions of that group.

For example:

Setuid: chmod u+s filename sets the setuid bit for the file.

Setgid: chmod g+s filename sets the setgid bit for the file.

Question 15: What is a “sticky-bit”?

Answer 15:
The sticky bit, in computing, serves as a permission flag applicable to both files and directories in Unix-like systems. Its function varies between files and directories.

For files, especially executables, the sticky bit once served a purpose by allowing the superuser to retain them in main memory even after their need ceased. This was done to minimize the need for swapping, ensuring faster access if the file was needed again. However, this function has become obsolete due to optimization in swapping techniques.

In the context of directories, setting the sticky bit impacts how files within the directory are handled. When the sticky bit is applied to a directory, the system treats the contained files differently. Specifically, only the file's owner, the directory's owner, or the root user can rename or delete the file. Without the sticky bit, users with write and execute permissions for the directory can modify or delete contained files, regardless of ownership. This setting is often applied to the /tmp directory, preventing ordinary users from altering or deleting other users' files.

The modern usage of the sticky bit primarily pertains to directories, protecting directory contents from unauthorized modification or deletion by non-owners. This security feature is prevalent in most contemporary Unix-like systems. For instance, in a shared directory like /tmp, files belong to individual owners, and non-owners are restricted from altering, overwriting, or renaming them.

Question 16: Use manual pages and explain what will command “uname -a” do?

Answer 16:
The uname command in Unix-like systems is used to display system information. When used with the -a or --all option, it prints comprehensive system details in a specified order, excluding specific details (-p and -i) if they are unknown:

Here's a breakdown of the information displayed by uname -a:

Kernel Name: Name of the operating system's kernel.

Network Node Hostname: The name of the machine on the network.

Kernel Release: Version number of the kernel.

Kernel Version: Additional version information of the kernel.

Machine Hardware Name: Identifies the machine's hardware or architecture.

Processor Type: Type of processor or architecture used.

Question 17: Use manual pages and explain what will command “wc -l” do?

Answer 17: wc - print newline, word, and byte counts for each file -l, --lines print the newline counts

The wc command in Unix-like systems is used to display counts of newline, word, and byte occurrences within files. When used with the -l or --lines option, wc specifically prints only the newline (line) counts for each file provided as input:

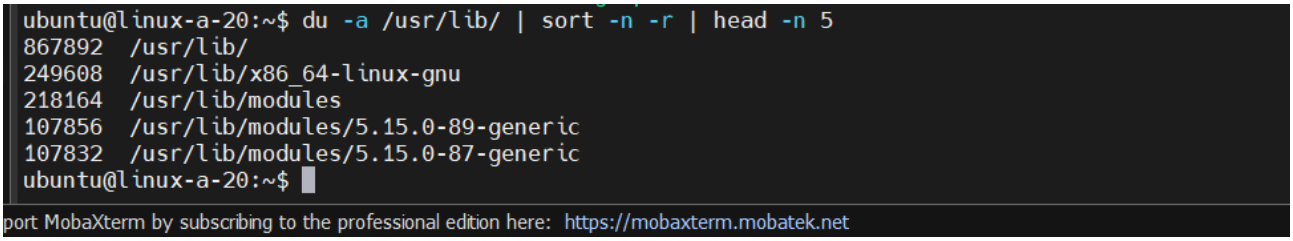
For instance, running wc -l filename or wc --lines filename will output only the count of newline characters in the file named "filename."

Question 18: Linux file handling intro

Answer 18:

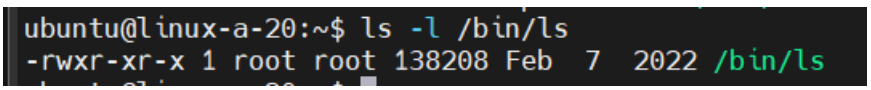
- **List five largest files in /usr/lib -directory:**

  du -a /usr/lib/ | sort -n -r | head -n 5



- **Find out what is the group for /bin/ls file?:**

  ls -l root



- **How do you change file or directory owner and group?:**

  chown

- **How do you change file permissions so that file user has all rights (read, write and execute), group and others have none?:**

sudo chmod 700 files

- **How do you change file permissions so that file user has read and write access (no execute), group and others have read access?:**

  sudo chmod 644 files

- **How do you change file permissions so that file user, group and others have only read and execute (no write) access?:**

  sudo chmod 555 files

- **Describe following file permissions and ownership:**
  drwxr-x--- 2 teemu root 4096 Jul 2 2002 webalizer

  Permissions: owner: read write execuate group:read execute others:none

  Ownership:teemu

- **Create directory "exercise1" under you home directory:**

```
ubuntu@linux-a-20:/home$ sudo mkdir exercise1
ubuntu@linux-a-20:/home$ ls
exercise1  ubuntu
ubuntu@linux-a-20:/home$ S
```

- **Create empty file (length 0 bytes) "qwerty.txt" to that directory:**

```
ubuntu@linux-a-20:/home/exercise1$ sudo touch qwerty.txt
ubuntu@linux-a-20:/home/exercise1$ ls
qwerty.txt
```

- **Change directory name "exercise1" to "exer2"?:**

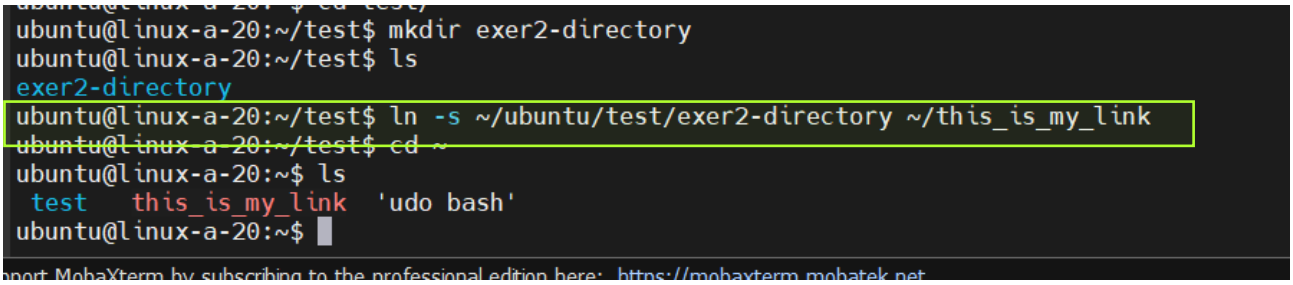
```
ubuntu@linux-a-20:/home$ mv exercise1 exer2
mv: cannot move 'exercise1' to 'exer2': Permission denied
ubuntu@linux-a-20:/home$ sudo mv exercise1 exer2
ubuntu@linux-a-20:/home$ ls
exer2  ubuntu
```

- **Change file qwerty.txt file permissions so that only you have just a read access to it and nothing else:**

```
ubuntu@linux-a-20:/home$ cd exer2
ubuntu@linux-a-20:/home/exer2$ chmod 400 qwerty.txt
chmod: changing permissions of 'qwerty.txt': Operation not permitted
ubuntu@linux-a-20:/home/exer2$ sudo chmod 400 qwerty.txt
ubuntu@linux-a-20:/home/exer2$ ls -l
total 0
-r-------- 1 root root 0 Nov  6 14:39 qwerty.txt
ubuntu@linux-a-20:/home/exer2$
```

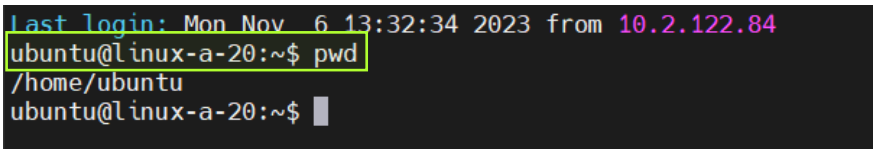
- **Create symbolic link to you home directory "this_is_my_link" and make it point to the exer2-directory:**

  ln -s ~/ubuntu/test/exer2-directory ~/this_is_my_link



- **How can you find out your current directory location and path? How far (in directories) are you from file system root?:**

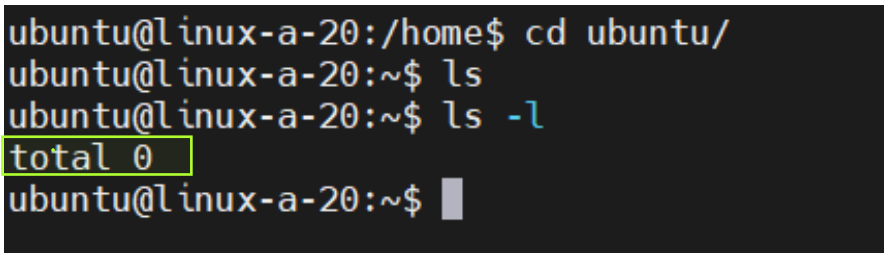
  Pwd : present working directory



Question 19: Remove files and directories which you created on this exercise

Answer 19 :

rm -r



Week 2

Questions 20: Watch some [live coding Twitch streams], select one and aswer:

Answer 20:

- **What is the programming language/languages used?**

  C++

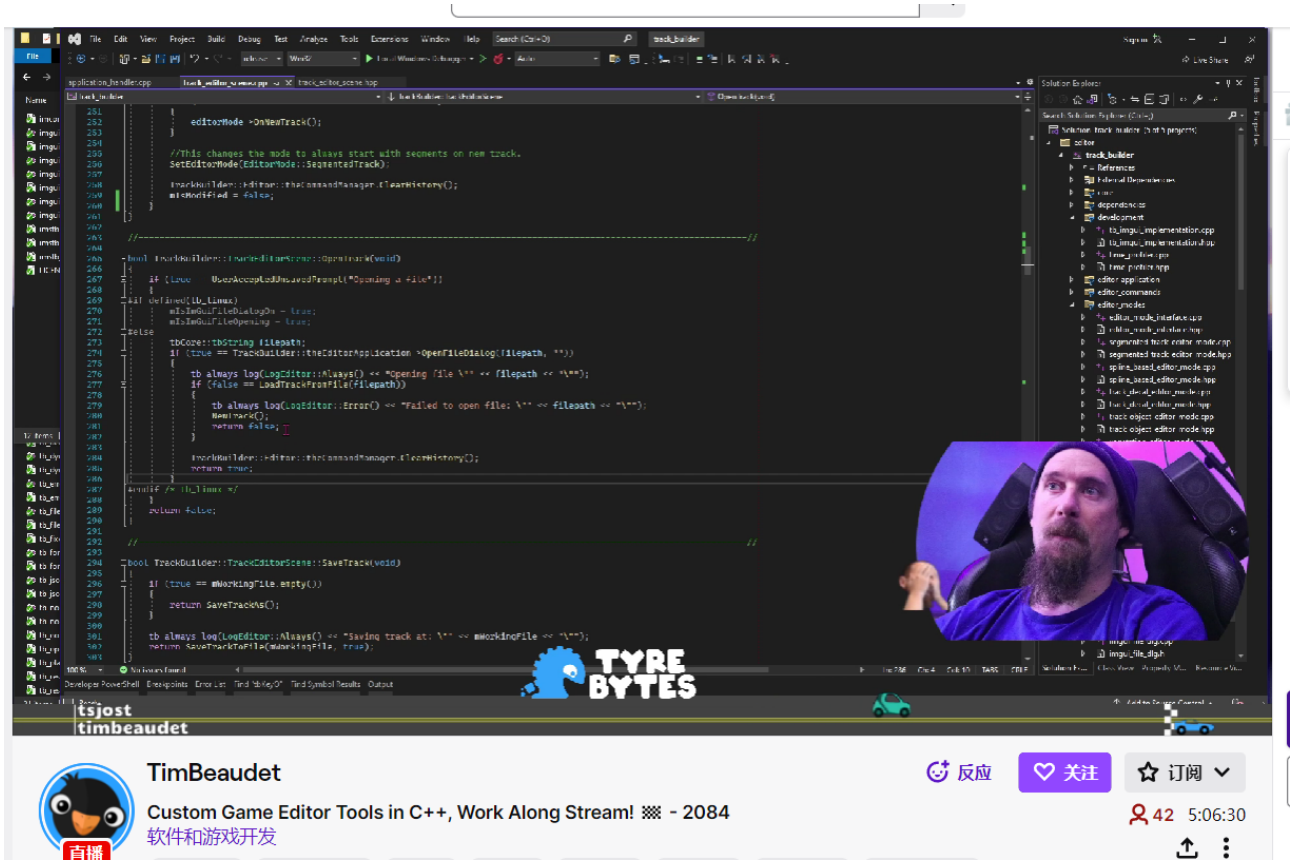
- **Which programming libraries and frameworks are being used?**

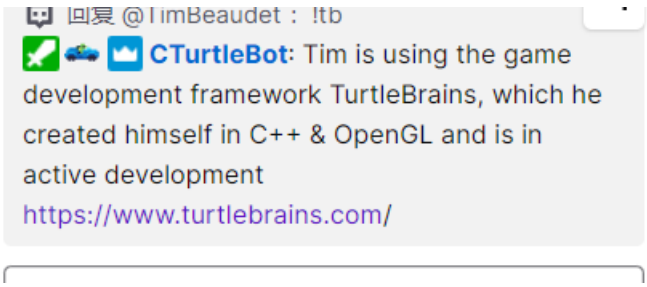
  Framework: TurtleBrains https://www.turtlebrains.com/documentation_0_3_5/

  OpenGL

- **What is the code editor / IDE being used?**

  Visual Studio

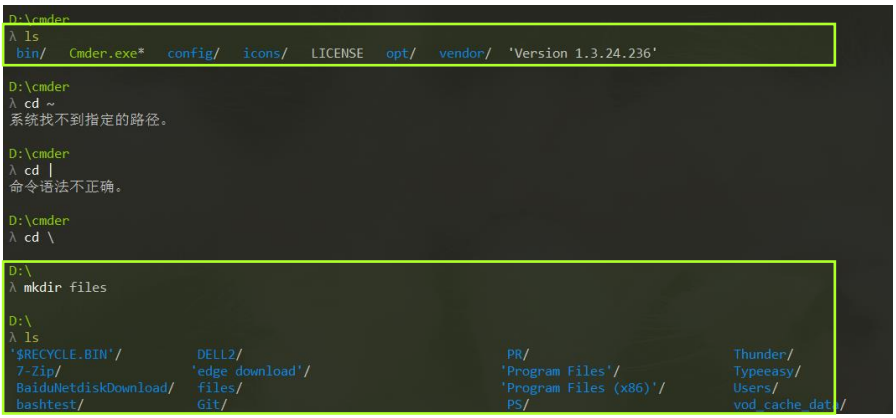




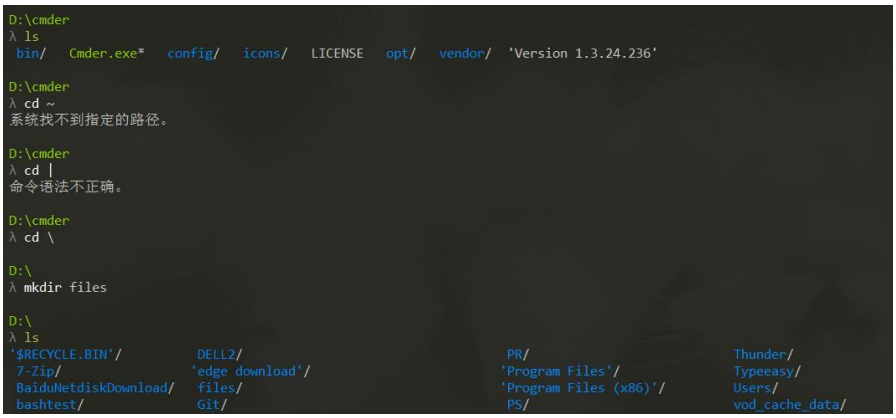
question21:If you have a Windows host:

Answer 21:

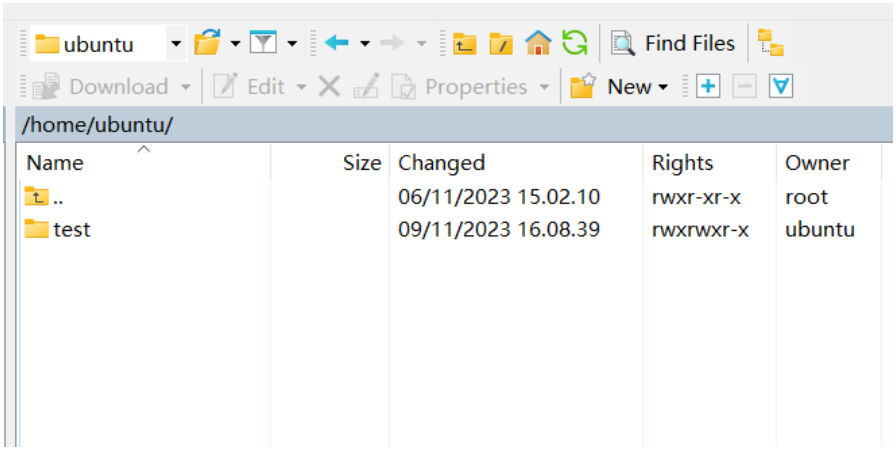
- **Install the Cmder and Winscp**



- **Try common file commands with Cmder (commands like ls, cd, mkdir, Bash shell etc)**



- **Use Winscp to move some files between the Windows and Linux hosts**



question 22:Study and explain shortly following commands and concepts

answer 22 :

- **zip, unzip**: zip - package and compress (archive) files

  unzip - list, test and extract compressed files in a ZIP archive

- **tar**: tar - an archiving utility
- **gzip**: gzip, gunzip, zcat - compress or expand files
- **xz:** xz, unxz, xzcat, lzma, unlzma, lzcat - Compress or decompress .xz and .lzma files
- **zcat, zgrep:** gzip, gunzip, zcat - compress or expand files

  zgrep - search possibly compressed files for a regular expression

- **compress**: compress: This term generally refers to the process of reducing the size of one or more files to save space on a storage device.
- **bzip2:** bzip2, bunzip2 - a block-sorting file compressor, v1.0.8
- **7z:** is a file archiver with a high compression ratio, capable of compressing files into the 7z format.
- **Ldd:** ldd - print shared object dependencies
- **gnu gcc / g++:**

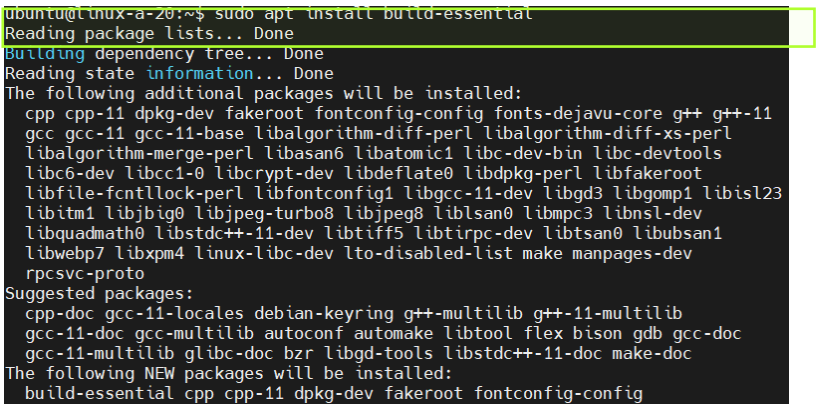
  GCC: Stands for GNU Compiler Collection. It is a suite of compilers for various programming languages, including C, C++, Objective-C, Fortran, Ada, and others. GCC can compile source code written in these languages into executable programs for different platforms.

  G++: Specifically, G++ is the GNU C++ compiler. It is a part of the GCC suite and is used for compiling C++ source code into executable programs.

- **Wget**: download link

question 23:Install build-essential meta package (containing development tools) to your server with: sudo apt install build-essential

Answer23:



question 24:Get the source code for curses-based (“text-graphics”) worm game [nibbles-1.2.tar.gz\]

answer 24:

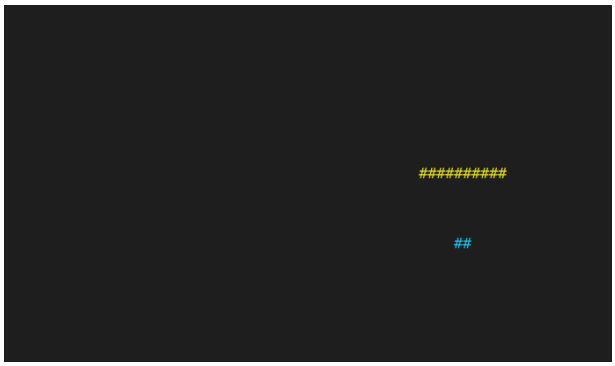
- **Unpack the source package to a some temporary directory under your home directory**

```
ubuntu@linux-a-20:~/test/nibbles-1.2$ cat Makefile
BIN             =       nibbles
OBJ             =       main.o misc.o screen.o
LDFLAGS =       -lncurses
CC              =       gcc -O2 -Wall

all: $(BIN)

clean:
        rm -f $(BIN) $(OBJ)

$(BIN): $(OBJ)
        $(CC) -o $(BIN) $(OBJ) $(LDFLAGS)
```

```
ubuntu@linux-a-20:~/test/nibbles-1.2$ make
gcc -O2 -Wall -c -o main.o main.c
main.c: In function 'main':
main.c:54:9: warning: implicit declaration of function 'memset' [-Wimplicit-func
tion-declaration]
   54 |         memset(grid,0,sizeof(grid));
      |         ^~~~~~
main.c:11:1: note: include '<string.h>' or provide a declaration of 'memset'
   10 | #include "screen.h"
  +++ |+#include <string.h>
   11 |
main.c:54:9: warning: incompatible implicit declaration of built-in function 'me
mset' [-Wbuiltin-declaration-mismatch]
   54 |         memset(grid,0,sizeof(grid));
      |         ^~~~~~
main.c:54:9: note: include '<string.h>' or provide a declaration of 'memset'
main.c:197:25: warning: implicit declaration of function 'memcpy' [-Wimplicit-fu
nction-declaration]
  197 |                         memcpy(hx, hx+HISTORYSIZE-length, length * sizeo
f(short int));
```

- **Compile the game and try playing it. Note: Ubuntu does not have ncursed development libraries installed by default. Use apt install to install the missing library dependencies: *sudo apt install libncurses-dev***

```
gcc -O2 -Wall -o nibbles main.o misc.o screen.o -lncurses
ubuntu@linux-a-20:~/test/nibbles-1.2$ ls
CHANGES   README    main.c  misc.c  misc.o   screen.c  screen.o
Makefile  config.h  main.o  misc.h  nibbles  screen.h
ubuntu@linux-a-20:~/test/nibbles-1.2$ ./nibbles
You crashed in a wall. Your score was 0.
ubuntu@linux-a-20:~/test/nibbles-1.2$ ./nibbles
You crashed in a wall. Your score was 0.
ubuntu@linux-a-20:~/test/nibbles-1.2$ ./nibbles
You crashed in a wall. Your score was 0.
ubuntu@linux-a-20:~/test/nibbles-1.2$ ./nibbles
You crashed in a wall. Your score was 2.
```



Questin25: Get the source code for another curses-based ("text-graphics") Tetris game [nct-1.4.tar.gz]

Answer 25:

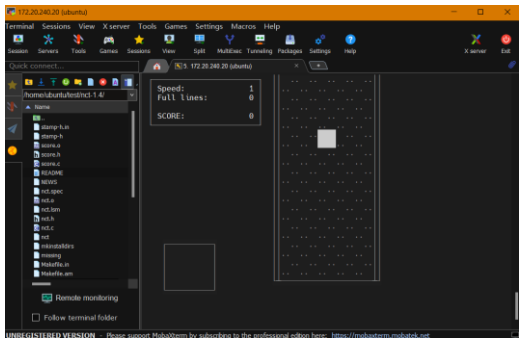
- **Unpack source package to temporary directory in your home directory.**
- **Use source package's configure script to generate Makefile with installation prefix pointing to your home directory**

```
ubuntu@linux-a-20:~/test/nct-1.4$ sh ./configure
creating cache ./config.cache
checking for a BSD compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking whether make sets ${MAKE}... yes
checking for working aclocal... missing
checking for working autoconf... missing
checking for working automake... missing
checking for working autoheader... missing
checking for working makeinfo... missing
checking for gcc... gcc
checking whether the C compiler (gcc -O  -s) works... yes
checking whether the C compiler (gcc -O  -s) is a cross-compiler... no
checking whether we are using GNU C... yes
checking whether gcc accepts -g... yes
```

- **Compile source code and install compiled files**

```
ubuntu@linux-a-20:~/test/nct-1.4$ make install
make[1]: Entering directory '/home/ubuntu/test/nct-1.4'
/bin/sh ./mkinstalldirs /usr/local/bin
  /usr/bin/install -c  nct /usr/local/bin/nct
/usr/bin/install: cannot create regular file '/usr/local/bin/nct': Permission de
nied
make[1]: *** [Makefile:164: install-binPROGRAMS] Error 1
make[1]: Leaving directory '/home/ubuntu/test/nct-1.4'
make: *** [Makefile:296: install-am] Error 2
ubuntu@linux-a-20:~/test/nct-1.4$ sudo blash
sudo: blash: command not found
ubuntu@linux-a-20:~/test/nct-1.4$ sodo bash
```

- **Test if game works**

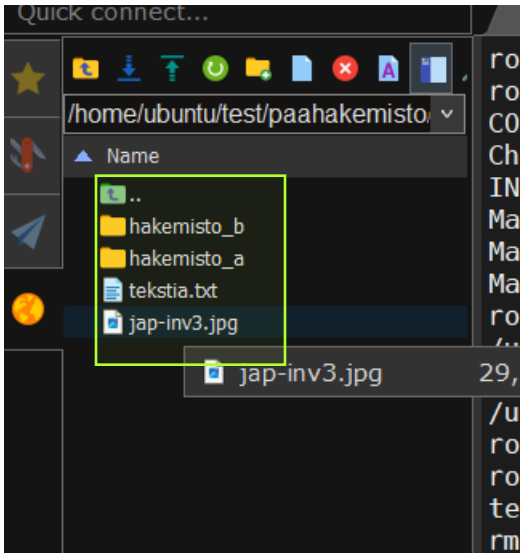
- 

- **Remove temporary game directory**

```
root@linux-a-20:/home/ubuntu/test/nct-1.4# make clean
test -z "nct" || rm -f nct
rm -f *.o core *.core
root@linux-a-20:/home/ubuntu/test/nct-1.4# make distclean
rm -f config.h
rm -f *.tab.c
rm -f TAGS ID
rm -f Makefile
rm -f config.cache config.log stamp-h stamp-h[0-9]*
test -z "confpaths.h" || rm -f confpaths.h
test -z "nct" || rm -f nct
rm -f *.o core *.core
rm -f config.status
root@linux-a-20:/home/ubuntu/test/nct-1.4#
```

Question 26: Download the file harj_zip_paketti.zip. Zip-package has following hierarchy

Answer 26:



Question 27: With the ZIP file:

- **Unpack package and all subdirectories to a temporary directory in your home directory**

```
ubuntu
ubuntu@linux-a-20:/home$ mkdir ~/temp_directory
ubuntu@linux-a-20:/home$ ls
ubuntu
ubuntu@linux-a-20:/home$ cd ubuntu/
ubuntu@linux-a-20:~$ ls
temp_directory  test
```

- **Create tar archive from unpacked files and directories and name it to a paketti.tar**

```
ubuntu@linux-a-20:~/temp_directory$ pwd
/home/ubuntu/temp_directory
ubuntu@linux-a-20:~/temp_directory$ unzip harj_zip_paketti.zip -d ~/temp_directo
ry
Archive:  harj_zip_paketti.zip
   creating: /home/ubuntu/temp_directory/paahakemisto/
   creating: /home/ubuntu/temp_directory/paahakemisto/hakemisto_a/
  inflating: /home/ubuntu/temp_directory/paahakemisto/hakemisto_a/karate_kat.jpg

  inflating: /home/ubuntu/temp_directory/paahakemisto/hakemisto_a/lazy.jpg
   creating: /home/ubuntu/temp_directory/paahakemisto/hakemisto_b/
  inflating: /home/ubuntu/temp_directory/paahakemisto/hakemisto_b/etherkill.jpg

  inflating: /home/ubuntu/temp_directory/paahakemisto/jap-inv3.jpg
  inflating: /home/ubuntu/temp_directory/paahakemisto/tekstia.txt
ubuntu@linux-a-20:~/temp_directory$ tar -cvf paketti.tar -C ~/temp_directory paa
hakemisto
paahakemisto/
paahakemisto/hakemisto_a/
paahakemisto/hakemisto_a/karate_kat.jpg
paahakemisto/hakemisto_a/lazy.jpg
paahakemisto/hakemisto_b/
paahakemisto/hakemisto_b/etherkill.jpg
paahakemisto/tekstia.txt
paahakemisto/jap-inv3.jpg
ubuntu@linux-a-20:~/temp_directory$
```

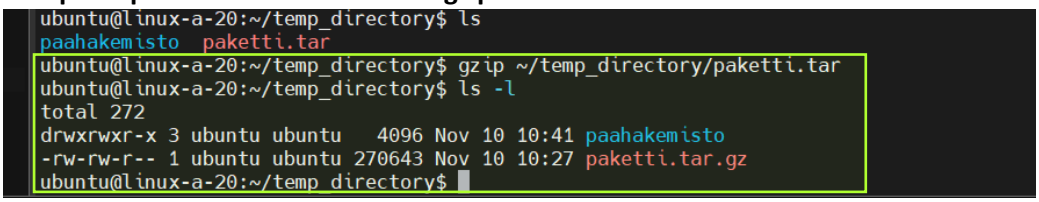
- **List contents of the paketti.tar. If everything is correct, delete paahakemisto directory and all subdirectories under it. Delete also the harj_zip_paketti.zip file. Don't delete the paketti.tar -file you just created.**

```
harj_zip_paketti.zip  paahakemisto  paketti.tar
ubuntu@linux-a-20:~/temp_directory$ tar -tvf paketti.tar
drwxr-xr-x ubuntu/ubuntu     0 2004-11-21 09:07 paahakemisto/
drwxr-xr-x ubuntu/ubuntu     0 2004-02-12 11:36 paahakemisto/hakemisto_a/
-rwxr--r-- ubuntu/ubuntu 30036 2004-02-12 11:34 paahakemisto/hakemisto_a/karate_kat.jpg
-rwxr--r-- ubuntu/ubuntu 68409 2004-02-12 11:34 paahakemisto/hakemisto_a/lazy.jpg
drwxr-xr-x ubuntu/ubuntu     0 2004-02-12 11:36 paahakemisto/hakemisto_b/
-rwxr--r-- ubuntu/ubuntu 95051 2004-02-12 11:35 paahakemisto/hakemisto_b/etherkill.jpg
-rw-rw-r-- ubuntu/ubuntu 154458 2004-11-21 09:07 paahakemisto/tekstia.txt
-rwxr--r-- ubuntu/ubuntu  29991 2004-02-12 11:34 paahakemisto/jap-inv3.jpg
ubuntu@linux-a-20:~/temp_directory$
```

- **Unpack only the etherkill.jpg file from tar archive.**

```
ubuntu@linux-a-20:~/temp_directory$ tar -xvf paketti.tar -C ~/temp_directory etherkill.jpg
tar: etherkill.jpg: Not found in archive
tar: Exiting with failure status due to previous errors
ubuntu@linux-a-20:~/temp_directory$ tar -xvf paketti.tar -C ~/temp_directory etherkill.jpg
tar: etherkill.jpg: Not found in archive
tar: Exiting with failure status due to previous errors
ubuntu@linux-a-20:~/temp_directory$ tar -xvf paketti.tar -C ~/temp_directory Etherkill.jpg
tar: Etherkill.jpg: Not found in archive
tar: Exiting with failure status due to previous errors
ubuntu@linux-a-20:~/temp_directory$ tar -tvf paketti.tar
drwxr-xr-x ubuntu/ubuntu     0 2004-11-21 09:07 paahakemisto/
drwxr-xr-x ubuntu/ubuntu     0 2004-02-12 11:36 paahakemisto/hakemisto_a/
-rwxr--r-- ubuntu/ubuntu 30036 2004-02-12 11:34 paahakemisto/hakemisto_a/karate_kat.jpg
-rwxr--r-- ubuntu/ubuntu 68409 2004-02-12 11:34 paahakemisto/hakemisto_a/lazy.jpg
drwxr-xr-x ubuntu/ubuntu     0 2004-02-12 11:36 paahakemisto/hakemisto_b/
-rwxr--r-- ubuntu/ubuntu 95051 2004-02-12 11:35 paahakemisto/hakemisto_b/etherkill.jpg
-rw-rw-r-- ubuntu/ubuntu 154458 2004-11-21 09:07 paahakemisto/tekstia.txt
-rwxr--r-- ubuntu/ubuntu  29991 2004-02-12 11:34 paahakemisto/jap-inv3.jpg
ubuntu@linux-a-20:~/temp_directory$ tar -xvf paketti.tar -C ~/temp_directory/paahakemisto/hakemisto_b/etherkill.jpg
tar: /home/ubuntu/temp_directory/paahakemisto/hakemisto_b/etherkill.jpg: Cannot open: No such file or directory
tar: Error is not recoverable: exiting now
ubuntu@linux-a-20:~/temp_directory$ tar -xvf paketti.tar etherkill.jpg -C ~/temp_directory
tar: etherkill.jpg: Not found in archive
tar: Exiting with failure status due to previous errors
ubuntu@linux-a-20:~/temp_directory$ tar -xvf paketti.tar -C ~/temp_directory paahakemisto/hakemisto_b/etherkill.jpg
paahakemisto/hakemisto_b/etherkill.jpg
ubuntu@linux-a-20:~/temp_directory$
```

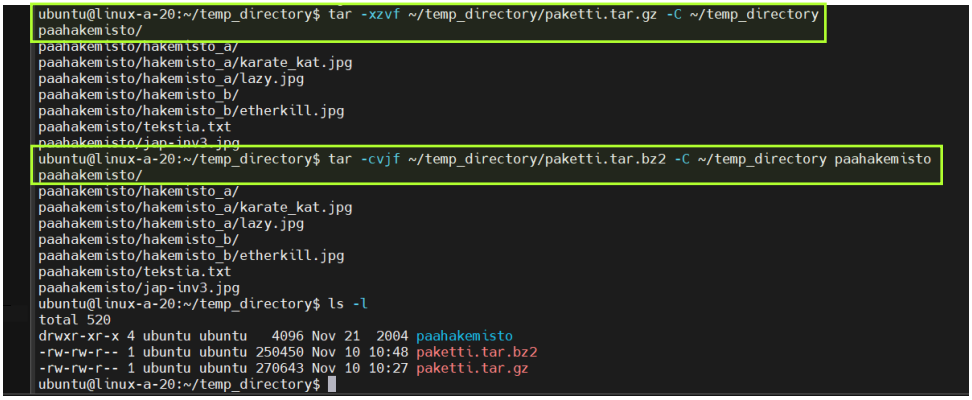
- **Compress paketti.tar archive with a gzip command.**



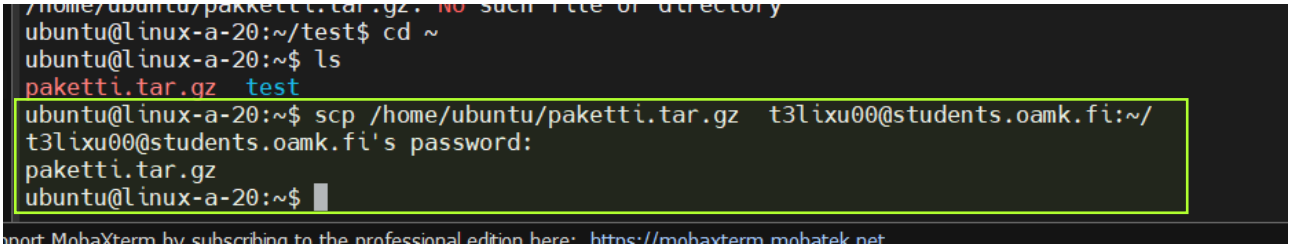
- **What is the size of paketti.tar.gz now?**

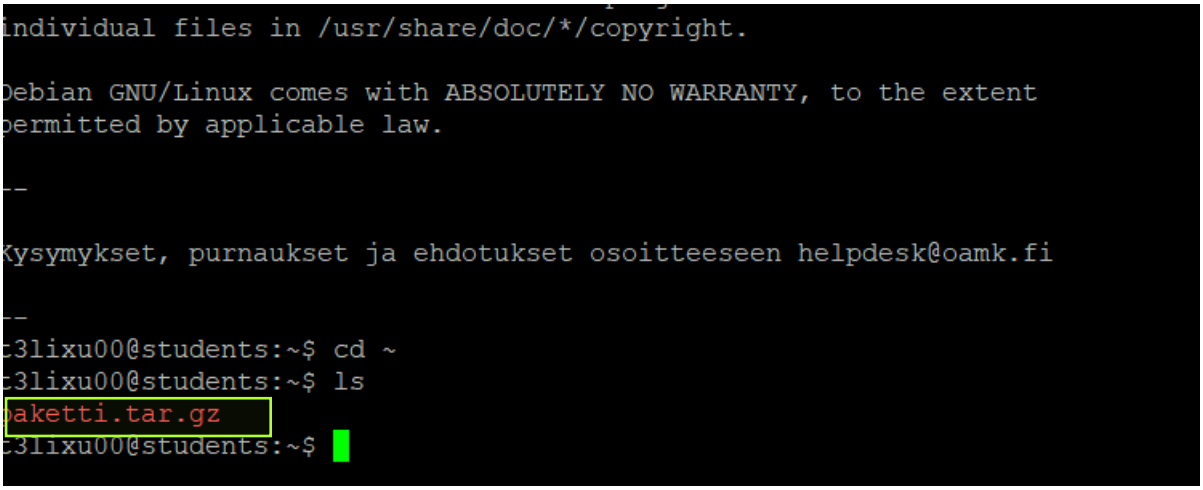
  270643

- **Uncompress paketti.tar.gz and compress it again, but now with bzip2. Check the size again. Any difference?**



- **Create some gzipped tar archive and use SSH (scp) to copy it to the students.oamk.fi (students.oamk.fi is a Linux server you can use with your Oamk user account credentials)**





- **Delete temporary files and directories created on this practice**

  done

Question 28: Compile this C source code with gcc and check if it works. helloworld.c source code:

Answer28:

```
ubuntu@linux-a-20:~/test$ sudo apt-get install gcc
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
gcc is already the newest version (4:11.2.0-1ubuntu1).
gcc set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 13 not upgraded.
ubuntu@linux-a-20:~/test$ touch ccode.c
ubuntu@linux-a-20:~/test$ ls
ccode.c
ubuntu@linux-a-20:~/test$ nano ccode.c
ubuntu@linux-a-20:~/test$ ls
ccode.c
ubuntu@linux-a-20:~/test$ cat ccode.c
#include <stdio.h>

int main(void) {
    printf("Hello, world!\n");
    return 0;
}
ubuntu@linux-a-20:~/test$ gcc -o helloworld ccode.c
ubuntu@linux-a-20:~/test$ ./helloworld
Hello, world!
ubuntu@linux-a-20:~/test$ ./helloworld
```

Question 29: Compile this C++ source code with g++ and test it. helloworld.cpp source code:

Answer29:

```
ubuntu@linux-a-20:~/test$ touch c++code.c++
ubuntu@linux-a-20:~/test$ ls
c++code.c++  ccode.c
ubuntu@linux-a-20:~/test$ nano c++code.c++
ubuntu@linux-a-20:~/test$
ubuntu@linux-a-20:~/test$ g++ -o helloworld c
/usr/bin/ld: cannot find c: No such file or directory
collect2: error: ld returned 1 exit status
ubuntu@linux-a-20:~/test$ g++ -o helloworld c++code.cpp
ubuntu@linux-a-20:~/test$
ubuntu@linux-a-20:~/test$ ./helloworld
Hello World!
ubuntu@linux-a-20:~/test$
```

Question30: With previously compiled helloworld C++ binary:

Answer 30:

- **What are statically linked libraries? Why would you use them?**

Statically linked libraries are those integrated into the compiled executable file itself during the compilation process. This integration includes all the necessary library code within the final executable. Consequently, when you execute a program, it operates independently without relying on external library files because it contains all essential components within the executable. However, statically linked binaries typically result in larger file sizes when compared to dynamically linked ones.

There are several compelling reasons to utilize statically linked libraries:

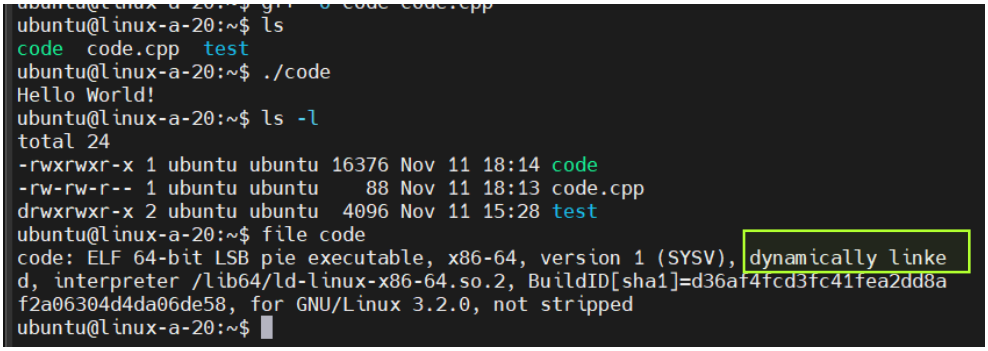
Portability: The executable file can be easily transferred to another system with the same architecture without concerns about differing library versions. It ensures consistent performance across compatible systems.

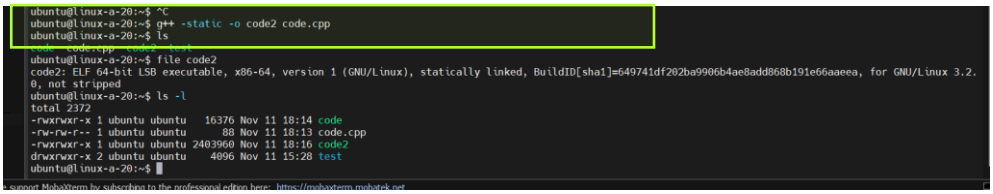
Distribution: Simplifies the distribution process of binaries since you don't need to ensure that the target system has the required shared libraries installed. This facilitates easier and more reliable deployment of applications.

Performance: Statically linked binaries may offer some performance advantages as they don't need to resolve dynamic libraries at runtime. This can result in faster application start-up times and reduced dependency on external resources during execution.

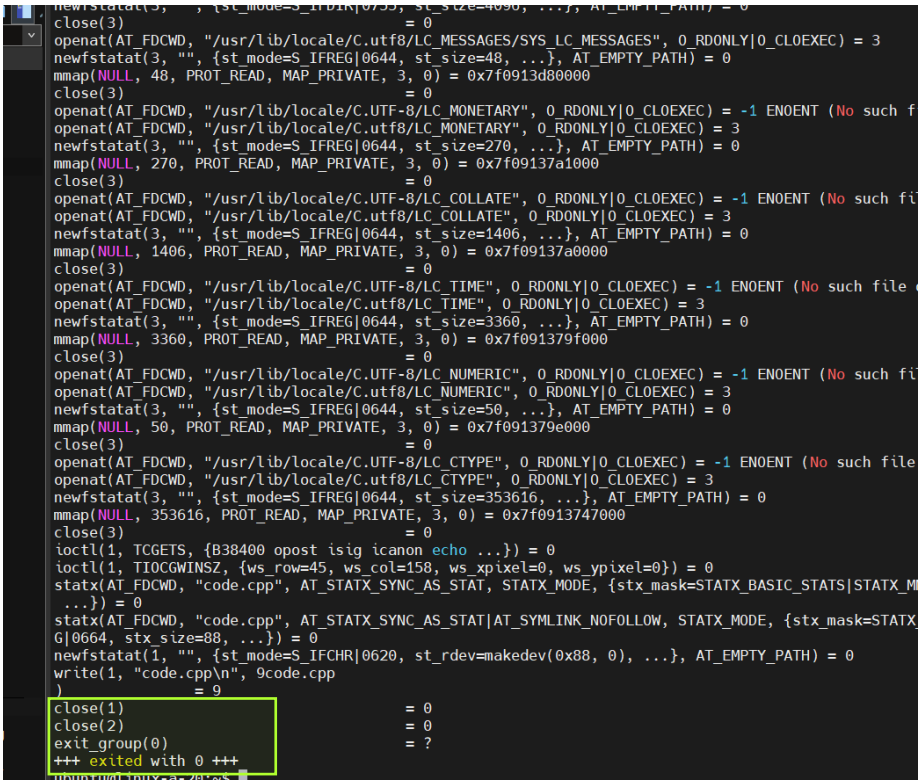
- **Inspect the size of ready binary file (that compiled helloworld binary). Compile it again and use some different output filename. With g++, use now statically linked libraries (with compiler's -static parameter). Compare the file sizes of statically and dynamically linked binaries**

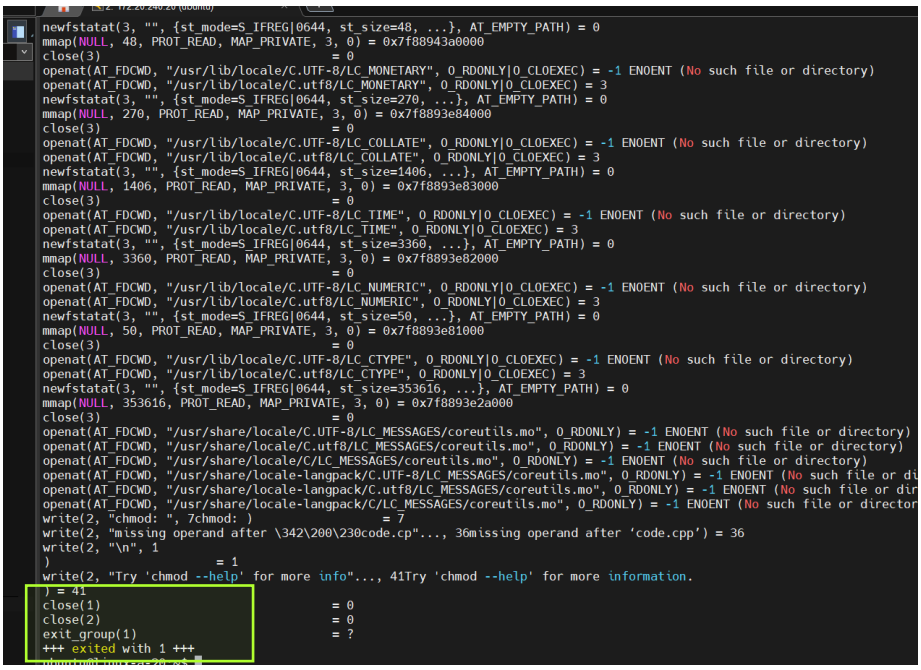
  Static 2403960 dymatically 16376





- **Use strace to inspect interiors (system calls) of ls command: "strace ls" and compare the output to a "strace chmod". Check _exit -values. Why chmod returns 1 and ls returns 0?**





ls returns 0 when it completes successfully without errors.

chmod returns a non-zero value (like 1) to indicate an error or failure if it encounters issues while modifying file permissions.

- **Why and when Unix administrators and programmers use system call tracing programs and debuggers such as gdb and strace?**

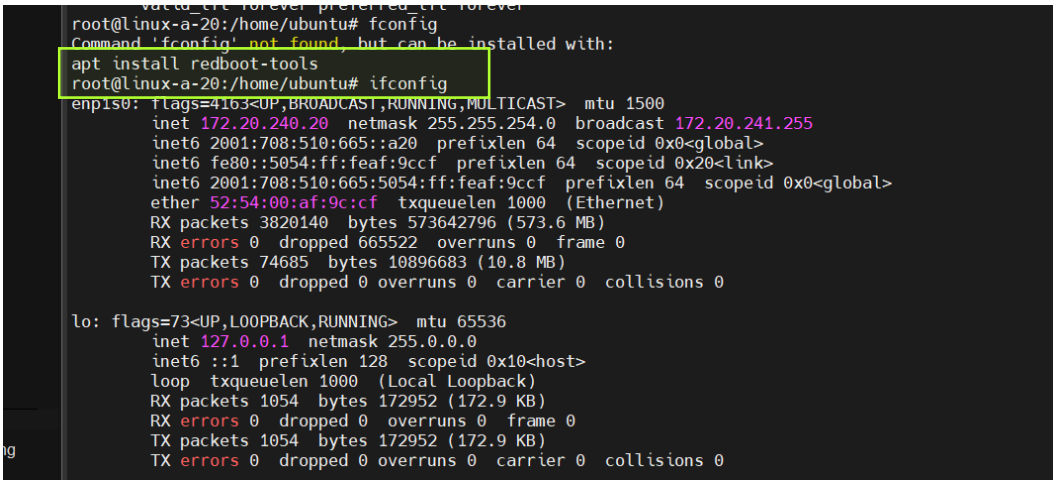
System call tracing programs like strace and debuggers such as gdb serve crucial roles for Unix administrators and programmers, primarily in troubleshooting, debugging, and comprehending program or operating system behaviors. Here are some key reasons and scenarios for utilizing these tools:

Debugging Programs: Developers employ debuggers like gdb to identify and rectify issues within their code. These tools enable inspection of program execution, setting breakpoints, variable examination, code stepping, and pinpointing logic errors or bugs.Tracing System Calls: Tools like strace monitor and trace system calls made by a program. This aids in understanding program-OS interactions, revealing performance bottlenecks, and identifying issues related to system calls (e.g., incorrect arguments, file access problems).Error Identification: When a program behaves unexpectedly or encounters errors, administrators and developers utilize these tools to trace execution, system calls, and resource interactions, facilitating identification of underlying issues.Performance Analysis: System call tracing tools help analyze program performance by tracking system calls, I/O operations, and resource utilization. This assists in code optimization, inefficient operation identification, and bottleneck detection affecting system performance.Understanding System Behavior: System call tracing tools assist administrators in gaining insights into system behavior, resource usage, monitoring system activities, and troubleshooting issues related to system services or daemons.Security Analysis: System call tracing aids in security-related investigations by monitoring system activities, detecting unauthorized access attempts, and identifying suspicious behavior within programs or the system.

Question 31: Solve these service management assignments (Note: most assignments will require root access):

Answer 31:

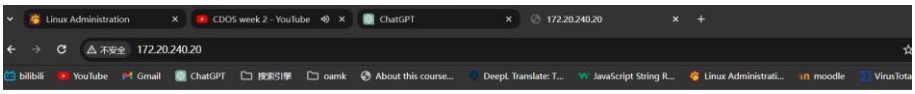
- **Check what network adapters your Linux host/server has with command: ip addr or ifconfig (ifconfig is not necessary installed by default)**



- **Listen inbound ICMP traffic in your server with tcpdump command line protocol analyzer and test if you can see the traffic when you ping your server: tcpdump -n -i YOUR_NETWORK_ADAPTER_NAME_HERE icmp**

```
root@linux-a-20:/home/ubuntu# tcpdump -n -i enp1s0 icmp
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp1s0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
18:48:03.411746 IP 10.2.122.84 > 172.20.240.20: ICMP echo request, id 1, seq 880, length 40
18:48:03.411989 IP 172.20.240.20 > 10.2.122.84: ICMP echo reply, id 1, seq 880, length 40
18:48:04.411966 IP 10.2.122.84 > 172.20.240.20: ICMP echo request, id 1, seq 881, length 40
18:48:04.412005 IP 172.20.240.20 > 10.2.122.84: ICMP echo reply, id 1, seq 881, length 40
18:48:05.425344 IP 10.2.122.84 > 172.20.240.20: ICMP echo request, id 1, seq 882, length 40
18:48:05.425385 IP 172.20.240.20 > 10.2.122.84: ICMP echo reply, id 1, seq 882, length 40
18:48:06.429016 IP 10.2.122.84 > 172.20.240.20: ICMP echo request, id 1, seq 883, length 40
18:48:06.429059 IP 172.20.240.20 > 10.2.122.84: ICMP echo reply, id 1, seq 883, length 40
^C
8 packets captured
8 packets received by filter
0 packets dropped by kernel
```

- **Install apache web server with apt install apache2 and test that you can access your server with a web browser**



hello there!

- **Listen TCP/80 (web) traffic in your server with tcpdump and test if you can see the inbound TCP SYN segments after you try to access your server with a web browser: tcpdump -n -i YOUR_NETWORK_ADAPTER_NAME_HERE tcp port 80**

```
root@linux-a-20:/home/ubuntu#  tcpdump -n -i enp1s0 tcp port 80
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp1s0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
18:50:44.017144 IP 10.2.122.84.7223 > 172.20.240.20.80: Flags [S], seq 3863064851, win 64240, options [mss 1361,nop,wscale 8,nop,nop,sackOK], length 0
18:50:44.017145 IP 10.2.122.84.7222 > 172.20.240.20.80: Flags [S], seq 3119108815, win 64240, options [mss 1361,nop,wscale 8,nop,nop,sackOK], length 0
18:50:44.017463 IP 172.20.240.20.80 > 10.2.122.84.7223: Flags [S.], seq 3315027518, ack 3863064852, win 64240, options [mss 1460,nop,nop,sackOK,nop,wscale 7],
 length 0
18:50:44.017530 IP 172.20.240.20.80 > 10.2.122.84.7222: Flags [S.], seq 3513013474, ack 3119108816, win 64240, options [mss 1460,nop,nop,sackOK,nop,wscale 7],
 length 0
18:50:44.097014 IP 10.2.122.84.7222 > 172.20.240.20.80: Flags [.], ack 1, win 1026, length 0
18:50:44.097014 IP 10.2.122.84.7223 > 172.20.240.20.80: Flags [.], ack 1, win 1026, length 0
18:51:15.504974 IP 172.20.240.20.80 > 10.2.122.84.7222: Flags [S.], seq 3513013474, ack 3119108816, win 64240, options [mss 1460,nop,nop,sackOK,nop,wscale 7],
 length 0
18:51:15.505023 IP 172.20.240.20.80 > 10.2.122.84.7223: Flags [S.], seq 3315027518, ack 3863064852, win 64240, options [mss 1460,nop,nop,sackOK,nop,wscale 7],
 length 0
18:51:15.599242 IP 10.2.122.84.7222 > 172.20.240.20.80: Flags [.], ack 1, win 1026, options [nop,nop,sack 1 {0:1}], length 0
18:51:15.599242 IP 10.2.122.84.7223 > 172.20.240.20.80: Flags [.], ack 1, win 1026, options [nop,nop,sack 1 {0:1}], length 0
18:51:29.070787 IP 10.2.122.84.7222 > 172.20.240.20.80: Flags [.], seq 0:1, ack 1, win 1026, length 1: HTTP
18:51:29.070788 IP 10.2.122.84.7223 > 172.20.240.20.80: Flags [.], seq 0:1, ack 1, win 1026, length 1: HTTP
18:51:29.070829 IP 172.20.240.20.80 > 10.2.122.84.7222: Flags [.], ack 1, win 502, options [nop,nop,sack 1 {0:1}], length 0
18:51:29.070851 IP 172.20.240.20.80 > 10.2.122.84.7223: Flags [.], ack 1, win 502, options [nop,nop,sack 1 {0:1}], length 0
18:51:35.601515 IP 172.20.240.20.80 > 10.2.122.84.7222: Flags [F.], seq 1, ack 1, win 502, length 0
18:51:35.609232 IP 172.20.240.20.80 > 10.2.122.84.7223: Flags [F.], seq 1, ack 1, win 502, length 0
18:51:35.761115 IP 10.2.122.84.7223 > 172.20.240.20.80: Flags [.], ack 2, win 1026, length 0
18:51:35.761116 IP 10.2.122.84.7222 > 172.20.240.20.80: Flags [.], ack 2, win 1026, length 0

^C
18 packets captured
18 packets received by filter
```

- **Explain what is runlevel?**

In Unix-like systems, a runlevel defines the system's operational state by specifying which services, daemons, and processes are running or permitted to run. Different runlevels correspond to distinct system configurations, commonly denoted by numerical identifiers (usually ranging from 0 to 6).

Here's an overview of traditional runlevels:

Runlevel 0: Initiates system shutdown or halts the system. It is responsible for commencing the shutdown procedure.

Runlevel 1: Also known as single-user mode or maintenance mode. It boots the system into a minimal state, beneficial for system maintenance tasks or troubleshooting, with limited services running.

Runlevels 2 to 5: These runlevels might have different uses across Unix-like systems. For instance, runlevel 3 might represent a multi-user mode without a graphical user interface (GUI), while runlevel 5 could include a GUI interface.

Runlevel 6: Reboots the system. This runlevel typically initiates a system reboot procedure.

- **Explain what is systemd?**
  Systemd is a Linux-based operating system's service and system manager. It's responsible for overseeing the system's initialization, managing services, and handling runtime system states. Developed as an alternative to the conventional System V init system, systemd offers advanced functionalities and improvements in managing system processes and services. Its primary tasks include streamlining system boot-up, controlling the lifecycle of services, and ensuring efficient management of system resources, contributing to better system performance and reliability.

- **Explain what are the files in /etc/init.d/ directory?**

  The /etc/init.d/ directory in Unix-like operating systems, such as Linux, contains scripts used by the System V init system for controlling and managing system services. These scripts are known as init scripts and are responsible for starting, stopping, restarting, and managing various daemons or services on the system.

  Here's an overview of the files and their purposes within the /etc/init.d/ directory:

  Service Init Scripts: Each file in this directory represents an init script associated with a specific system service or daemon. These scripts typically follow a naming convention related to the service they control (e.g., apache2, mysql, ssh, etc.).

  System Service Control: Init scripts provide standardized commands to control services. Common commands include start, stop, restart, reload, status, and force-reload. For example:

  sudo /etc/init.d/apache2 start: Initiates the Apache HTTP Server service.

  sudo /etc/init.d/mysql stop: Stops the MySQL database service.

  Initialization and Shutdown: During system startup or shutdown, the System V init system runs these scripts to start or stop services according to the runlevel or system state. Each script contains instructions on how a specific service should be handled during system initialization or shutdown.

  Symbolic Links in Runlevel Directories: Additionally, symbolic links to these init scripts are placed in the runlevel directories (/etc/rc*.d/). These links determine which services should be started or stopped when the system enters a specific runlevel.

  For instance, within /etc/rc3.d/ (runlevel 3), you might find symbolic links pointing to init scripts located in /etc/init.d/. These links define which services should be started or stopped when the system boots into runlevel 3.

- **Study but don’t do: What is runlevel 6? What is the purpose of init 6 command? How would you do the same with systemd?**

  Runlevel 6: Reboot the system. Entering runlevel 6 typically initiates a system reboot.

  The init 6 command in such systems is used to transition the system to runlevel 6, initiating the reboot process. When you execute init 6 from the command line, it signals the init process to change the runlevel to 6, leading to the orderly shutdown of services and subsequent reboot of the system.

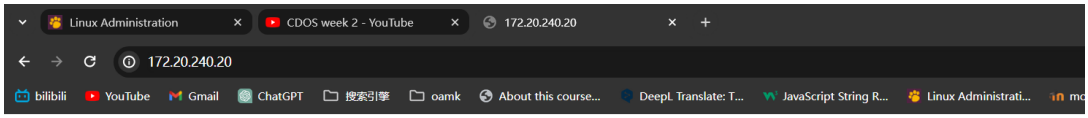
  With systemd, the equivalent command to reboot the system is:

  systemctl reboot

Question 32: Test these service management commands with your web server and use web browser to verify the operation whether the server is running or not:

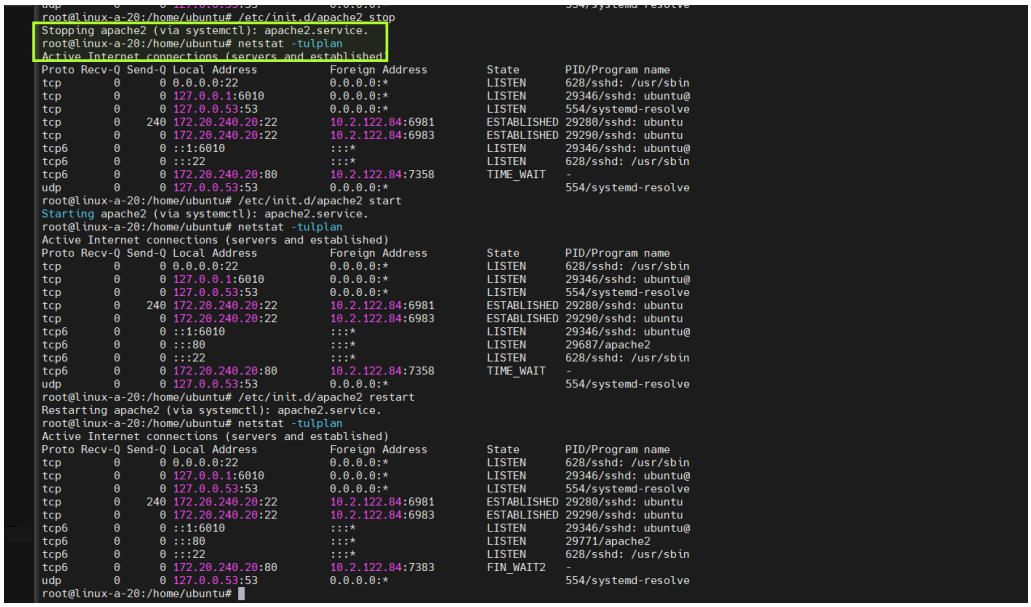
Answer32:

```
root@linux-a-20:/home/ubuntu# service apache2 stop
root@linux-a-20:/home/ubuntu# netstat -tulplan
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      628/sshd: /usr/sbin
tcp        0      0 127.0.0.1:6010          0.0.0.0:*               LISTEN      29346/sshd: ubuntu@
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN      554/systemd-resolve
tcp        0    272 172.20.240.20:22        10.2.122.84:6981        ESTABLISHED 29280/sshd: ubuntu
tcp        0      0 172.20.240.20:22        10.2.122.84:6983        ESTABLISHED 29290/sshd: ubuntu
tcp6       0      0 ::1:6010                :::*                    LISTEN      29346/sshd: ubuntu@
tcp6       0      0 :::22                   :::*                    LISTEN      628/sshd: /usr/sbin
udp        0      0 127.0.0.53:53           0.0.0.0:*                           554/systemd-resolve
root@linux-a-20:/home/ubuntu# service apache2 stop
root@linux-a-20:/home/ubuntu# netstat -tulplan
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      628/sshd: /usr/sbin
tcp        0      0 127.0.0.1:6010          0.0.0.0:*               LISTEN      29346/sshd: ubuntu@
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN      554/systemd-resolve
tcp        0    240 172.20.240.20:22        10.2.122.84:6981        ESTABLISHED 29280/sshd: ubuntu
tcp        0      0 172.20.240.20:22        10.2.122.84:6983        ESTABLISHED 29290/sshd: ubuntu
tcp6       0      0 ::1:6010                :::*                    LISTEN      29346/sshd: ubuntu@
tcp6       0      0 :::22                   :::*                    LISTEN      628/sshd: /usr/sbin
udp        0      0 127.0.0.53:53           0.0.0.0:*                           554/systemd-resolve
root@linux-a-20:/home/ubuntu# service apache2 restart
root@linux-a-20:/home/ubuntu# netstat -tulplan
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      628/sshd: /usr/sbin
tcp        0      0 127.0.0.1:6010          0.0.0.0:*               LISTEN      29346/sshd: ubuntu@
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN      554/systemd-resolve
tcp        0     48 172.20.240.20:22        10.2.122.84:6981        ESTABLISHED 29280/sshd: ubuntu
tcp        0      0 172.20.240.20:22        10.2.122.84:6983        ESTABLISHED 29290/sshd: ubuntu
tcp6       0      0 ::1:6010                :::*                    LISTEN      29346/sshd: ubuntu@
tcp6       0      0 :::80                   :::*                    LISTEN      29594/apache2
tcp6       0      0 :::22                   :::*                    LISTEN      628/sshd: /usr/sbin
udp        0      0 127.0.0.53:53           0.0.0.0:*                           554/systemd-resolve
root@linux-a-20:/home/ubuntu#
```



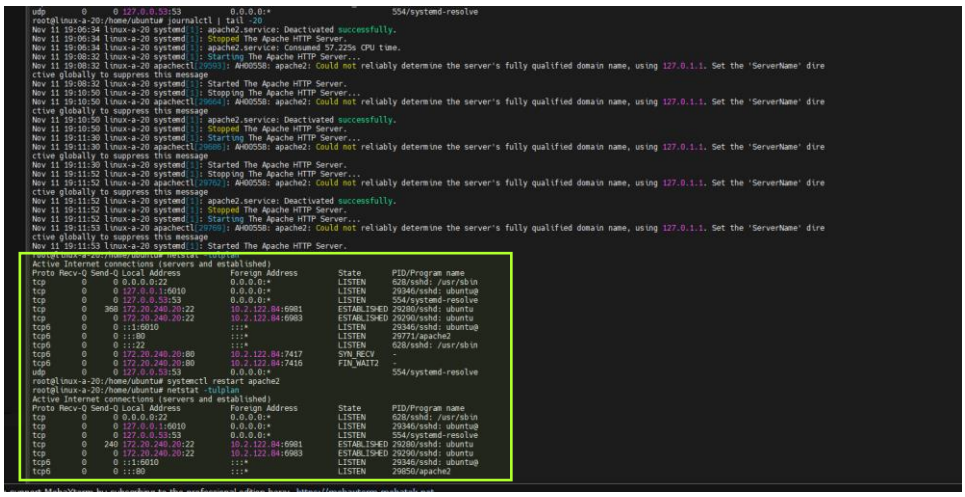
Question33: Test these service management commands with your web server and use web browser to verify the operation whether the server is running or not:

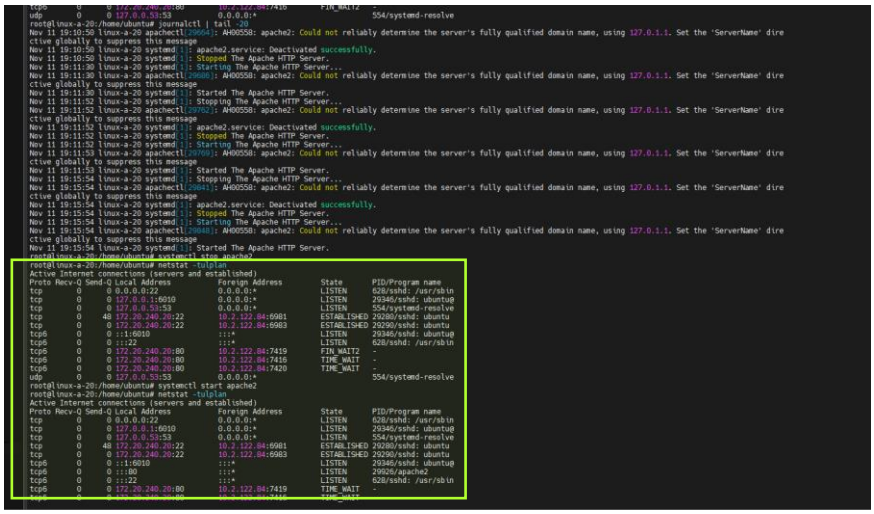
Answer33:



Question34: Test these service management commands with your web server and use web browser to verify the operation whether the server is running or not. Try and explain:

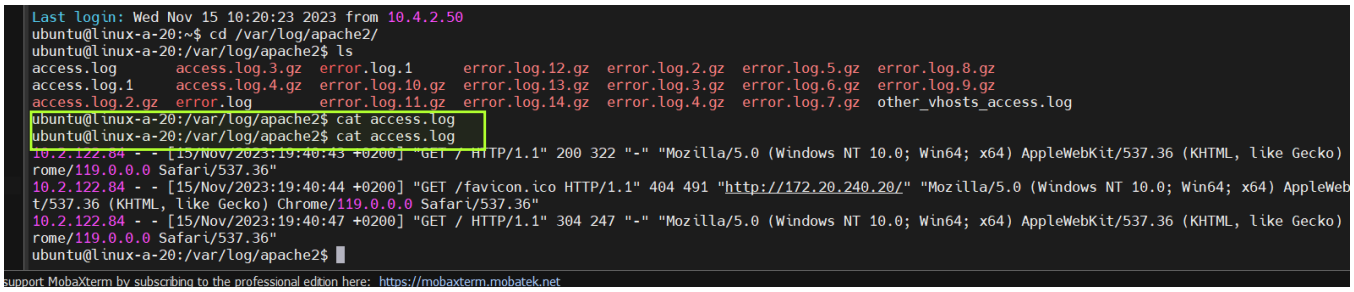
Answer34:run run run run close





Question35: Check Apache access.log file contents in /var/log/apache2/ directory. Can you find your connections to the web server?

Answer35: cat access.log



Week 3

Question 36 : Study and explain shortly following commands and concepts:

Answer 36:

- **sh, bash, zsh**

  sh dash —sh (Bourne Shell): It's the earliest command-line shell in Unix systems, offering a relatively simple syntax. It provides basic command interpretation and control structures, serving as the ancestor to many modern shells.

  bash bash (Bourne Again Shell): It's an extension of the Bourne Shell with additional features such as command completion, aliases, conditional testing, and more. It's the default shell in many Linux systems.

  zsh (Z Shell): It's an improved version of bash, offering more functionalities like enhanced auto-completion, greater flexibility in configuration options, and extensive plugin support. It's a powerful shell choice, especially for users requiring advanced features and personalized customization.

- **screen and tmux**

  Screen is a full-screen window manager that multiplexes a physical terminal between several processes (typically interactive shells)

  tmux is a terminal multiplexer: it enables a number of terminals to be created, accessed, and controlled from a single screen. tmux may be detached from a screen and continue running in the background, then later reattached.

- **ps, pgrep, pstree, pidof**

  ps report a snapshot of the current processes.

  pgrep, pkill, pidwait - look up, signal, or wait for processes based on name and other attributes

  pstree - display a tree of processes

  pidof -- find the process ID of a running program.

- **jobs, disown**

  jobs: The 'jobs' command displays a list of jobs that are currently running in the background or stopped. When you execute multiple commands in a terminal, some of them might be running in the background (using '&' at the end of the command) or have been suspended (using Ctrl + Z). 'jobs' helps you see a list of these background or stopped jobs, along with their status and job numbers.

  disown: The 'disown' command is used to remove jobs from the shell's job table. When you run a process in a terminal, it's associated with the shell session. Using 'disown' with a specific job number or PID removes that job from the shell's job table, allowing it to continue running even after you close the terminal. This effectively detaches the job from the terminal session, preventing it from receiving any further SIGHUP (hang-up) signals.

- **fg, bg**

fg (Foreground): This command brings a background process into the foreground. When you run a command in a terminal, it typically runs in the foreground, and you can see its output directly. However, if you suspend a process (by pressing Ctrl + Z), it goes into the background. Using 'fg' followed by a job number or a process ID (PID) will move that job or process back to the foreground, allowing you to interact with it directly.

bg (Background): This command moves a suspended or stopped process into the background, allowing it to continue executing but not taking control of the terminal. After suspending a process using Ctrl + Z, you can use 'bg' followed by the job number or PID to resume the process in the background.

- **top, htop**

  top - display Linux processes The top program provides a dynamic real-time view of a running system.

  It can display system summary information as well as a list ofprocesses or threads currently being managed by the Linux kernel.

  htop - interactive process viewer

  It is similar to top, but allows you to scroll vertically and horizontally, and interact using a pointing device (mouse).

- **nice, renice**

  nice - run a program with modified scheduling priority

  renice - alter priority of running processes

- **su, sudo**

  su - run a command with substitute user and group ID

  sudo, sudoedit — execute a command as another user

- **sleep**

  sleep - delay for a specified amount of time

- **xargs**

  xargs - build and execute command lines from standard input

- **nohup**

  nohup - run a command immune to hangups, with output to **a non-tty**

- **kill**

  kill - send a signal to a process

- **pkill, killall**

  pgrep, pkill, pidwait - look up, signal, or wait for processes based on

  name and other attributes

  killall - kill processes by name

- **w, who**

w - Show who is logged on and what they are doing.

who - show who is logged on

- **write, wall**

  write - send a message to another user

  wall - write a message to all users

- **aliases**

  Aliases are a feature in Unix-like operating systems (such as Linux) that allow users to create custom shortcuts or alternative names for commands or command sequences. They're essentially a way to create your own shorthand for longer or frequently used commands, making the terminal more efficient and user-friendly.

- **source, .bashrc**

  source: 'source' is a shell built-in command used to execute commands from a file within the current shell session. When you use 'source' followed by a filename, it reads and executes commands from that file in the current shell context. It's often used to update the current shell's environment with settings or variables defined in the specified file.

  .bashrc: This is a configuration file for the Bash shell (Bourne Again SHell). It stands for "Bash run commands". The '.bashrc' file is typically located in the user's home directory (~/.bashrc) and is executed automatically when a new interactive Bash shell is started. It contains shell configuration settings, aliases, environment variables, and other customizations for the Bash shell.

- **shell build-in variables, export**

  Shell Built-in Variables: These variables are inherent to the shell environment and provide information or configuration details that the shell uses. Some examples of built-in variables include:

  - $HOME: Represents the user's home directory.
  - $PATH: Specifies the directories where executable files are located.
  - $PWD: Stores the current working directory.
  - $SHELL: Indicates the path to the current shell executable.
  - $USER: Contains the username of the current user.

These variables are predefined and are utilized by the shell for various operations and user interactions.

export Command: In shell scripting, the 'export' command is used to set an environment variable. When a variable is exported, it becomes available to child processes or scripts that are executed from the current shell session.

Question 37: How and when you start new shells? How to exit a shell?

Answer 37:

Shells are interfaces that allow users to interact with operating systems through commands

Launch Command: Once the terminal is open, you can start a new shell by simply typing the name of the shell you want to use. For example:

- bash (Bourne Again SHell)
- zsh (Z Shell)
- fish (Friendly Interactive SHell)
- sh (Bourne Shell)
- csh (C Shell)
- ksh (Korn Shell)

Press Enter: Hit the Enter key after typing the name of the shell. This will start a new shell session.

Exiting a Shell:

- To exit a shell and return to the previous environment or close the terminal window:
- Exit Command: Typing the exit command and then pressing Enter will close the shell session. This command works across most shell environments.
- Keyboard Shortcut: Sometimes, pressing Ctrl + D (EOF - End of File) also exits the shell.

Question 38:Add shell alias “diskusage” to your shell startup-files (example .bashrc). Alias should print only current disk usage of your home directory

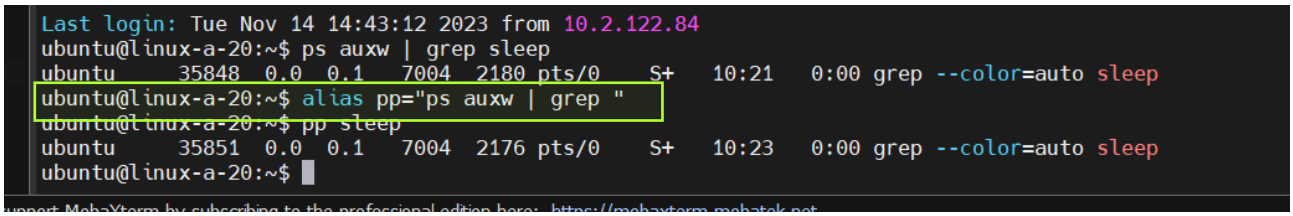
Answer 38:

alias diskusage="du ~/"

```
ubuntu@linux-a-20:~$ LS
LS: command not found
ubuntu@linux-a-20:~$ alias diskusage="du ~/"
ubuntu@linux-a-20:~$ dis
diskusage    disown       distro-info
ubuntu@linux-a-20:~$ diskusage
4       /home/ubuntu/test
4       /home/ubuntu/.cache
8       /home/ubuntu/.ssh
4       /home/ubuntu/.local/share/nano
8       /home/ubuntu/.local/share
12      /home/ubuntu/.local
4       /home/ubuntu/.config/procps
8       /home/ubuntu/.config
124     /home/ubuntu/
ubuntu@linux-a-20:~$
```

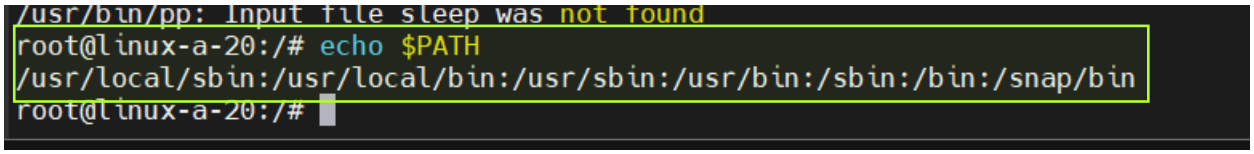
Question 39:Create shell alias “pp” which requires one parameter and will print all running processes including details with that name. Usage example:

Answer 39:



Question 40: Which directories are currently in you PATH variable?

Answer 40: echo $PATH



question41:How do you start process directly into background when entering a command?

Answer 41:

To start a process directly in the background when entering a command in a Unix-based terminal, you can append an ampersand (&) at the end of the command. This symbol tells the shell to initiate the process in the background, allowing you to continue using the terminal for other tasks without waiting for the process to finish.

For instance, if you want to start a process named example_process in the background

example_process &

ctrl c cancel

ctrl z stop

Question 42:Start few sleep 60 processes (one minute idle loop) to the background and:

Answer 42:

- **How can you find and terminate them all with one-liner? Try not to use pkill, killall or xargs -commands.**

  kill $(ps aux | grep '[s]leep 60' | awk '{print $2}')

  Let's break down this command:ps aux lists all running processes.grep '[s]leep 60' searches for the sleep 60 processes. Using [s]leep in the grep command prevents it from matching the actual grep process itself in the output.awk '{print $2}' extracts the second column from the output of grep, which contains the process IDs (PIDs) of the sleep 60 processes.kill is used to terminate processes based on their PIDs. The $(...) syntax captures the output of the preceding command and provides it as arguments to kill.Running this command will find all the sleep 60 processes running in the background and terminate them by sending the default SIGTERM signal.

- **How would you do the previous killing task with xargs?**

```
ubuntu@linux-a-20:~$ sleep 60&
[1] 36298
ubuntu@linux-a-20:~$ sleep 60&
[2] 36299
ubuntu@linux-a-20:~$ sleep 60&
[3] 36300
ubuntu@linux-a-20:~$ jobs -p | xargs kill
[1]   Terminated              sleep 60
[2]-  Terminated              sleep 60
[3]+  Terminated              sleep 60
ubuntu@linux-a-20:~$ jobs
```

- **Start one 1000 second sleep to the foreground.**

```
ubuntu@linux-a-20:~$ sleep 1000
^Z
[1]+  Stopped                 sleep 1000
ubuntu@linux-a-20:~$
```

- **How do you suspend it?**

  Ctrl + z suspend

  Crtl + c cancel

- **How do you list current jobs?**

```
^Z
[1]+  Stopped                 sleep 1000
ubuntu@linux-a-20:~$ jobs
[1]+  Stopped                 sleep 1000
ubuntu@linux-a-20:~$
```

- **How do you get previous sleep process back to foreground?**

```
ubuntu@linux-a-20:~$ jobs
[1]   Running                 sleep 1000 &
[2]-  Running                 sleep 1000 &
[3]+  Stopped                 sleep 1000
ubuntu@linux-a-20:~$ fg 1
sleep 1000
^Z
[1]+  Stopped                 sleep 1000
ubuntu@linux-a-20:~$
```

- **Suspend process again and send it to background.**

```
[1]+  Stopped                 sleep 1000
ubuntu@linux-a-20:~$ bg 1
[1]+ sleep 1000 &
ubuntu@linux-a-20:~$ jobs
[1]   Running                 sleep 1000 &
[2]-  Running                 sleep 1000 &
[3]+  Stopped                 sleep 1000
ubuntu@linux-a-20:~$
```

- **Kill previous sleep process from background.**

```
ubuntu@linux-a-20:~$ jobs
[2]+  Running                 sleep 1000 &
ubuntu@linux-a-20:~$ ps
    PID TTY          TIME CMD
  36461 pts/0    00:00:00 bash
  36478 pts/0    00:00:00 sleep
  36572 pts/0    00:00:00 ps
ubuntu@linux-a-20:~$ kill PID 36478
-bash: kill: PID: arguments must be process or job IDs
[2]+  Terminated              sleep 1000
ubuntu@linux-a-20:~$
```

Question 43: What is the difference between kill -9 and kill -1?

Answer 43:

kill -9 is a forceful termination that immediately stops the process without allowing it to perform any cleanup actions. It's often used as a last resort to terminate unresponsive or problematic processes.

kill -1 sends a signal that can be caught and interpreted by the process, often used to instruct a process to reload its configuration or restart gracefully without abruptly terminating it.

Question 44: Delete unnecessary files created in this practice.

Answer 44:done

Week 4

Question 45: Study and explain shortly following commands and concepts:

Answer 45 :

- **cat, tac**

  cat - concatenate files and print on the standard output

  tac - concatenate and print files in reverse

- **grep / egrep**

  grep, egrep, fgrep, rgrep - print lines that match patterns

- **wc**

  wc - print newline, word, and byte counts for each file

- **sort**

  sort - sort lines of text files

- **cut**

  cut - remove sections from each line of files

- **awk**

  gawk - pattern scanning and processing language

- **sed**

  sed - stream editor for filtering and transforming text

- **tr**

  tr - translate or delete characters

- **expand, unexpand**

  expand - convert tabs to spaces

  unexpand - convert spaces to tabs

- **uniq**

  uniq - report or omit repeated lineshead

- **tail**

  tail - output the last part of files

- **echo**

  echo - display a line of text

- **column**

  column - columnate lists

- **fold**

  fold - wrap each input line to fit in specified width

- **join**

  join - join lines of two files on a common field

- **paste**

  paste - merge lines of files

- **tee**

  tee - read from standard input and write to standard output and files

- **nl**

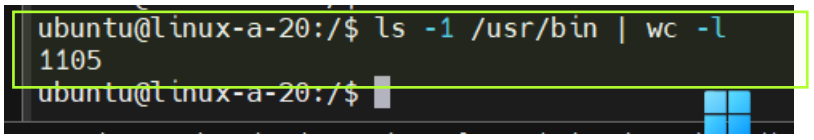
  nl - number lines of files

- **shul**

  prints lines by random order

Question 46: Use word counter and piping to count how many files or directories are in /usr/bin -directory?

Answer 46:



ls -1 /usr/bin | wc -l

Here's a breakdown of the command:

ls -1 /usr/bin: Lists the contents of the /usr/bin directory with one entry per line (-1 means used to display each entry on a new line).

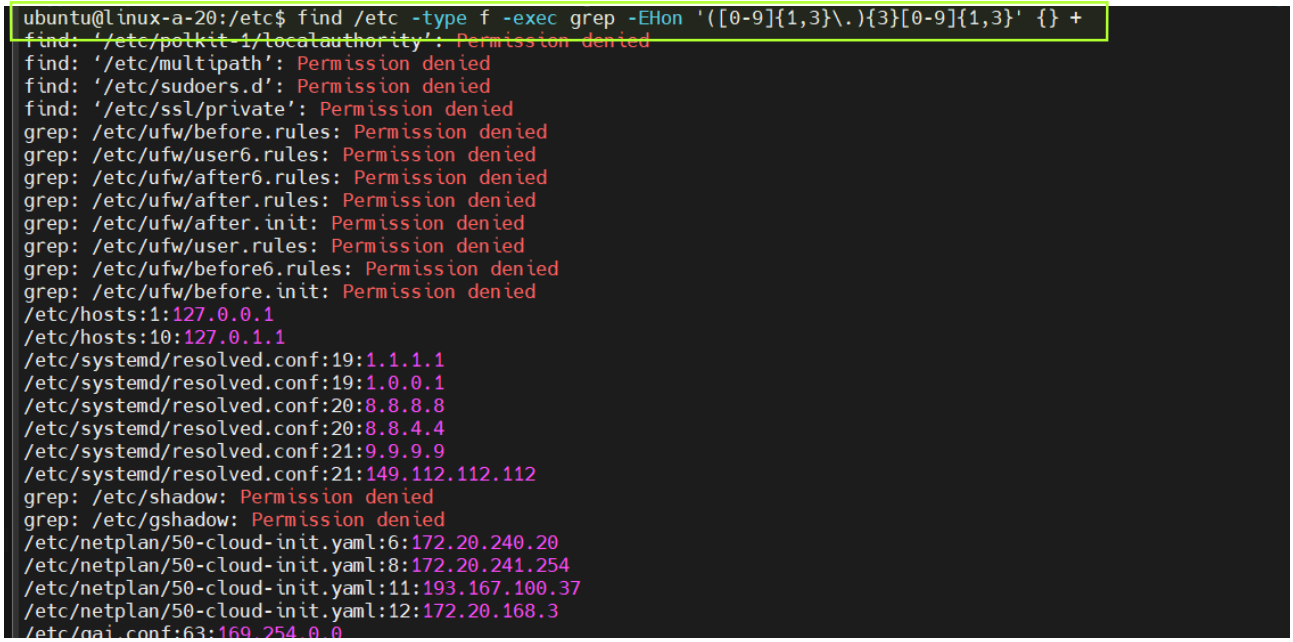
|: Pipes the output of the ls command to the next command.

wc -l: Performs a word count on the input received from ls and counts the number of lines (-l option).
The wc command stands for "word count" in Unix and Linux systems. The -l option used with wc is used to count the number of lines in the input.

Question 47: Use grep and extended regular expression syntax to list all files from /etc directory recursively which have IPv4 addresses mentioned inside.

Answer 47:

find /etc -type f -exec grep -EHon '([0-9]{1,3}\.){3}[0-9]{1,3}' {} +

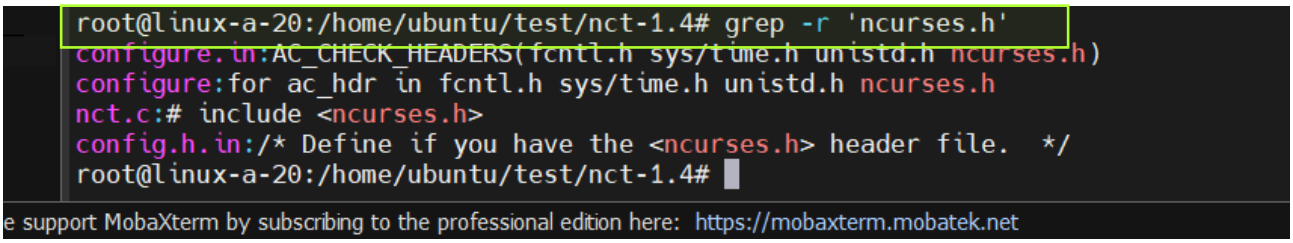


Question 48: Download and extract Tetris game source file nct-1.4.tar.gz and perform specific tasks with grep, wc, sorting, and data filtering commands.

Answer 48:

- **Use grep to find which files contain string ncurses.h**

  /home/ubuntu/test/nct-1.4# grep -r 'ncurses.h'

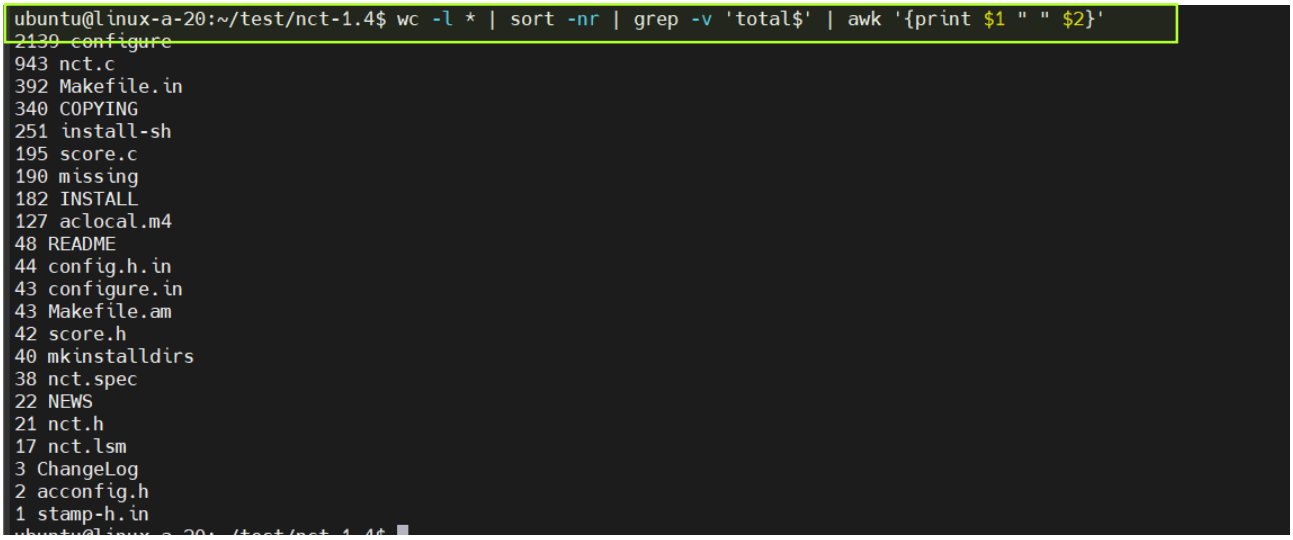


  -r, --recursive

  Read all files under each directory, recursively, following symbolic links only if they are on the command line. Note that if no file operand is given, B<grep> searches the working directory. This is equivalent to the -d recurse option.

- **Use wc command to list line counts of each file and sort the output from longest to shortest file. Use data filtering command to remove the total amount of lines line in the beginning of sort output. Final result should be something like this**

  ~/test/nct-1.4$ wc -l * | sort -nr | grep -v 'total$' | awk '{print $1 " " $2}'



Question 49: Use wget to download irclog.txt and answer specific questions related to it.

Answer 49:

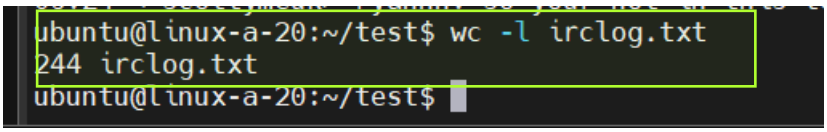
- **How many lines are in the file?**

  244

  ~/test$ wc -l irclog.txt

  Here are some common options used with wc:

  -l: Counts the number of lines in a file.

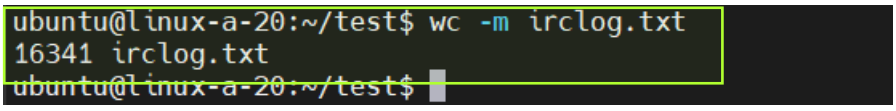


- **How many characters are in the file?**

  16341

  wc -m irclog.txt

  -m: Counts the number of characters in a file (for some Asian languages, where a character can be multiple bytes).



- **List only lines where the timestamp starts with 05 and save the output to a file called result.txt**

grep '^05' irclog.txt > result.txt

```
ubuntu@linux-a-20:~/test$ grep '^05' irclog.txt > result.txt
ubuntu@linux-a-20:~/test$
```

- **Print result.txt in reverse order**

  Tac result.txt

```
ubuntu@linux-a-20:~/test$ tac result.txt
05:59 < ryann> var lsd
05:59 < ryann> manager/controllers/Signup.cfc:         var lsd = query("getLinodeSignupData", "SELECT FieldName, Fieldvalue FROM ln_LinodeSignupData WHERE Lin
odeSignupID = #ls.LinodeSignupID#").recordSet;
05:59 < ryann> I like how linode does stuff like this
05:58 < AlexC_> ryann: People have been bugging them to upgrade the forums for a long time
05:57 < ryann> It's so dirty I feel bad reading it
05:57 < ryann> this code
05:57 < ryann> <cfif ListLen(cgi.script_name, "/") gt 2 AND ListGetAt(cgi.script_name, 2, "/") eq "linode" AND NOT ListFind("index.cfm,linode_edit.cfm,linode_
resize.cfm,label.cfm,cancel.cfm,dc_choose.cfm,su.cfm,pastdue.cfm", ListGetAt(cgi.script_name, 3, "/"))>         <cfinclude template="/members/linode/common/ds
p_topNav.cfm"> </cfif>
05:57 < ryann> phpbb2
05:57 < ryann> Ruchira, the forum is pretty old too
05:56 < Ruchira> ryann: the shore was abandoned long time ago. Im wondering why would they use that host name for a db host
05:56 < ryann> Their current source is horrible to read trough
05:56 < ryann> gdi can't linode just use some normal language
05:53 < ryann> $dbpasswd = 'cfr41qa';
05:53 < ryann> $dbuser = 'linode';
05:53 < ryann> $dbname = 'linode_forums';
05:53 < ryann> $dbhost = 'newnova.theshore.net';
05:53 < ryann> btw
```

- **Create numerical statistics from the irclog.txt file: How many lines each nickname wrote. Use only those lines where someone actually said something and ignore the all other lines. Output should be something like this:**

```
ubuntu@linux-a-10:~/regex_examples$ cat irclog.txt | grep -v '\-!\-' | cut -d"<" -f2 | cut -d">" -f1 | sort | uniq -c
| sort -n -r
     44 ryan_
     41 ryan||
     34 ryann
     15 AlexC_
     12 scottymeuk
     12 Ruchira
     11 ryannn
     11 kyhwana
     11 gerryvdm_mbp
     10 shmoon
      8 chesty
      6 Ruchira_
      3 ssthormess
      2 rww
      2 mestri
      2 drclawski
      1 ryan|
      1 mikegrb
      1 gkmngrgn
      1 d-b
      1 akerl
ubuntu@linux-a-10:~/regex_examples$
```

Question 50: List only 5 largest files from /usr/bin -directory

Answer 50:

- **Print largest files first**

  ls -S -l /usr/bin |head -n 5

```
ubuntu@linux-a-20:/$ ls -S -l /usr/bin |head -n 6
total 157348
-rwxr-xr-x 1 root root    24887296 May 13  2023 x86_64-linux-gnu-lto-dump-11
-rwxr-xr-x 1 root root    16138392 May 29 15:08 snap
-rwxr-xr-x 1 root root     5913032 Jun 11 08:26 python3.10
-rwxr-xr-x 1 root root     4420608 Aug 18  2022 batcat
-rwxr-xr-x 2 root root     3798008 May 23  2023 perl
```

- **Try to not use the ls command's -S option but use use sort command (and related text processing commands if necessary)**

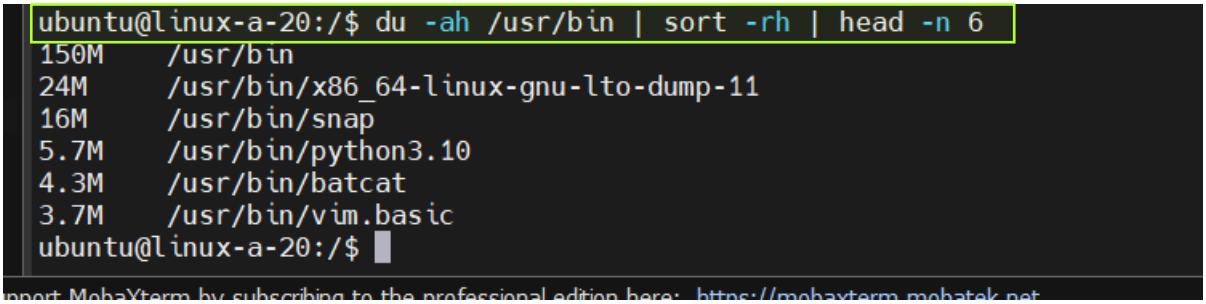
  du -ah /usr/bin | sort -rh | head -n 6

  Explanation:

  du -ah /usr/bin: Lists sizes of all files and directories in /usr/bin, with human-readable sizes (-h option) and including individual files (-a option).

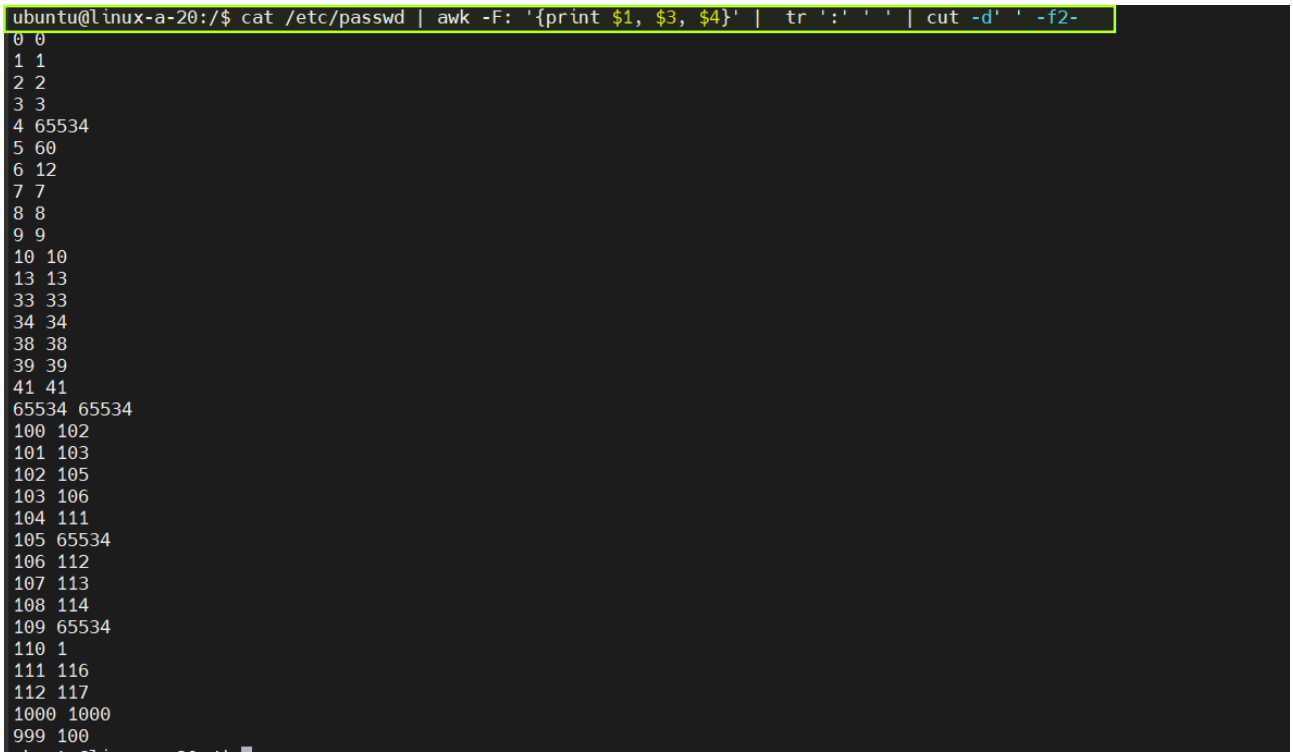
  sort -rh: Sorts the output numerically (-n) in reverse (-r) order, considering human-readable file sizes (-h).

  head -n 5: Displays the top 5 entries (i.e., the 5 largest files) from the sorted list.



Question 51:Print only usernames, UID and GID numbers from /etc/passwd -file. Replace all colons with a whitespace. Redirect output to file a “users.txt” in your home directory. Tip: In this example line from /etc/passwd the UID = 101 and GID = 50:

Answer 51: cat /etc/passwd | awk -F: '{print $1, $3, $4}' | tr ':' ' ' | cut -d' ' -f2-

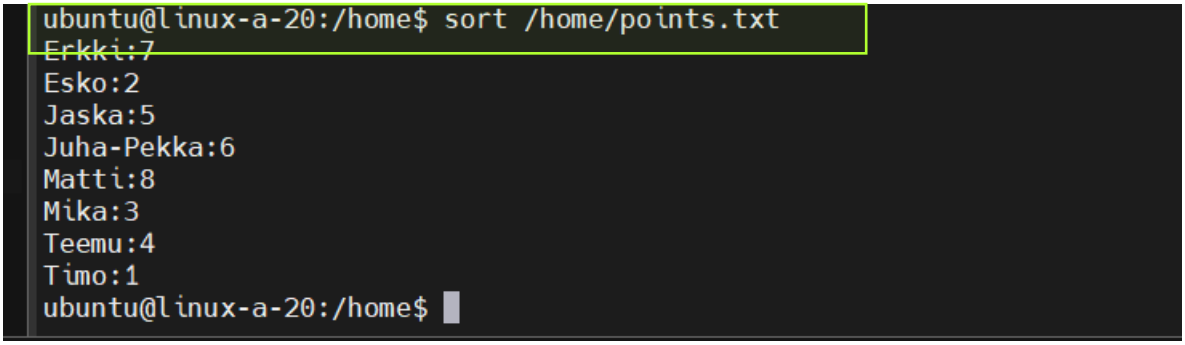


Question 52: Use text editor nano to create a points.txt file to your home directory with following content. This list presents first names and some game scores. Who has most points, wins

Answer 52: sorted list in the terminal (STDOUT)

- **List contents of points.txt in alphabetic order to STDOUT**

  sort /home/points.txt



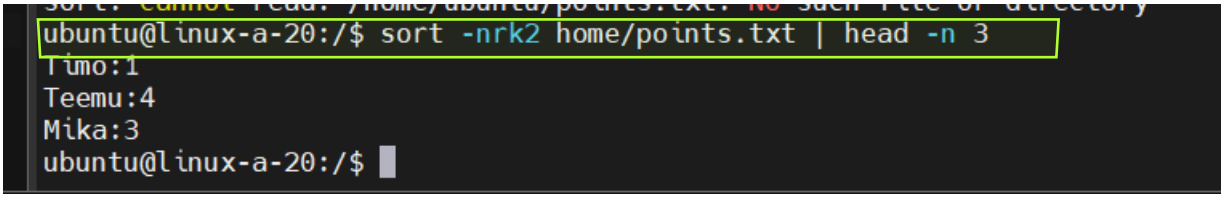
- **List contens of file on to STDOUT, but now order is score based. List only best three players with most points**

  sort -nrk2 home/points.txt | head -n 3

  This command does the following:

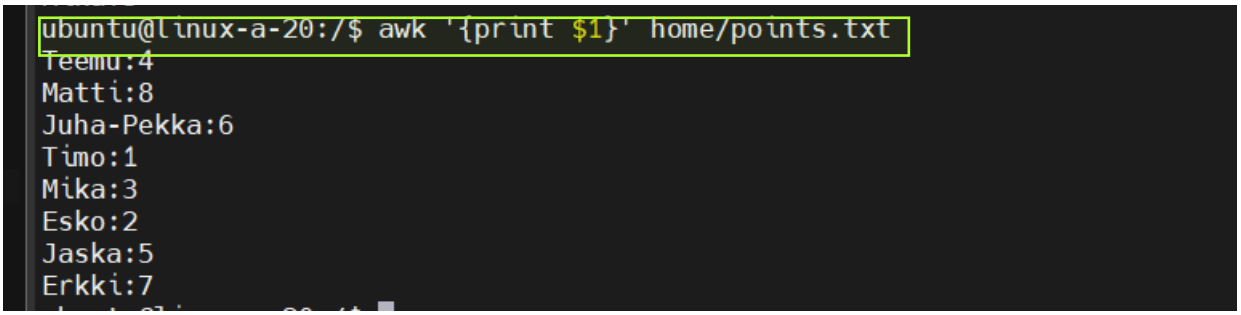
  sort -nrk2 ~/points.txt: Sorts the points.txt file numerically (-n) and in reverse (-r) order based on the second column (-k2), which contains the scores.

  head -n 3: Displays only the first three lines (top three players with the highest scores) from the sorted output.



- **How do you list only player names and filter all other data**

  awk '{print $1}' home/points.txt



- **List only first three characters from the beginning of each line of points.txt**

  cut -c 1-3 home/points.txt

```
ubuntu@linux-a-20:/$ cut -c 1-3 home/points.txt
Tee
Mat
Juh
Tim
Mik
Esk
Jas
Erk
```

- **List points.txt but translate all characters to upper-case**

  tr '[:lower:]' '[:upper:]' < home/points.txt

```
ubuntu@linux-a-20:/$ tr '[:lower:]' '[:upper:]' < home/points.txt
TEEMU:4
MATTI:8
JUHA-PEKKA:6
TIMO:1
MIKA:3
ESKO:2
JASKA:5
ERKKI:7
```

- **List points.txt so that points are printed before names**

  sort -n points.txt | sed 's/:/ /'| awk '{ print $2 " " $1 }' | sort -r

```
ubuntu@linux-a-20:/home$ sort -n points.txt | sed 's/:/ /'| awk '{ print $2 " " $1 }' | sort -r
8 Matti
7 Erkki
6 Juha-Pekka
5 Jaska
4 Teemu
3 Mika
2 Esko
1 Timo
ubuntu@linux-a-20:/home$
```

- **Sort points.txt in alphabetic order and add line numbers in front of lines**

  sort home/points.txt | nl

```
ubuntu@linux-a-20:/$ sort home/points.txt | nl
     1  Erkki:7
     2  Esko:2
     3  Jaska:5
     4  Juha-Pekka:6
     5  Matti:8
     6  Mika:3
     7  Teemu:4
     8  Timo:1
```

Question 53: How do you list last 5 lines from the /etc/passwd file?

Answer 53:

tail -n 5 /etc/passwd

```
ubuntu@linux-a-20:/$ tail -n 5 /etc/passwd
pollinate:x:110:1::/var/cache/pollinate:/bin/false
landscape:x:111:116::/var/lib/landscape:/usr/sbin/nologin
fwupd-refresh:x:112:117:fwupd-refresh user,,,:/run/systemd:/usr/sbin/nologin
ubuntu:x:1000:1000:Ubuntu:/home/ubuntu:/bin/bash
lxd:x:999:100::/var/snap/lxd/common/lxd:/bin/false
```

Question 54: How do you list first 5 lines from the /etc/passwd file?

Answer 54:

head -n 5 /etc/passwd

```
ubuntu@linux-a-20:/$ head -n 5 /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
```

Question 55: What does tail -f filename command do?

Answer 55:

tail - output the last part of files -f, --follow[={name|descriptor}]output appended data as the file grows;an absent option argument means 'descriptor'

Question 56: Fetch current weather in Oulu with lynx (TIP: if there is no lynx, install it with: sudo apt install lynx). The command to download Oulu's weather data is: lynx -dump 'https://opendata.fmi.fi/wfs?request=getFeature&storedquery_id=fmi::observations::weather::timevaluepair&place=oulu&timestep=10&parameters=temperature'

Answer 56:

- **Filter the output so that only temperature is displayed and nothing else**

  lynx - a general purpose distributed information browser for the World Wide Web

```
ubuntu@linux-a-10:~$ lynx -width=50 -dump 'https://opendata.fmi.fi/wfs?request=getFeature&storedquery_id=fmi::observations::weather::timevaluepair&place=oulu&timestep=10&parameters=temperature'
```

Question 57: Use wget to download this stock market textfile

Answer 57:

- **Use grep (or egrep) and regular expressions to list only companies with "I" anywhere in in code part.**

```
ubuntu@linux-a-10:~$ cat stocks.txt | grep ':.*I.*:'
Fiskars Corporation :FISAS: -0,36% 8,35 8,39 8,44 8,37 8,37
KCI Konecranes Plc :KCI1V: +0,06% 34,10 34,16 34,20 34,05 34,17
```

- **List (only) company names and stock values starting with character "M".**

```
ubuntu@linux-a-10:~$ cat stocks.txt | grep '^M.*:'
Metso Corporation :MEO1V: -0,08% 11,77 11,79 11,80 11,73 11,79
M-real Corporation A :MRLAV: -0,42% 4,64 4,83 4,75 4,75 4,75
M-real Corporation B :MRLBV: -1,06% 4,65 4,67 4,75 4,64 4,67
```

- **Print line only if the company name begins with a character "R" and last stock value is 8,xx**

```
ubuntu@linux-a-10:~$ cat stocks.txt | grep '^R.*8,..$'
Rautaruukki Corporation :RTRKS: +0,24% 8,46 8,50 8,52 8,42
Rocla Oyj :ROC1V: -0,60% 8,20 8,25 8,25 8,20 8,25
```

- **List all companies except the names starting with characters "R" or "W"**

```
ubuntu@linux-a-20:~/test$ cat stocks.txt | grep '^[RW]'
Rautaruukki Corporation :RTRKS: +0,24% 8,46 8,50 8,52 8,42
Raute Plc A :RUTAV: -1,13% 7,82 7,90 7,90 7,85 7,90 7,89
Rocla Oyj :ROC1V: -0,60% 8,20 8,25 8,25 8,20 8,25
Wärtsilä Corporation A :WRTAV: +3,47% 16,82 17,00 17,00 16,74 17,00
Wärtsilä Corporation B :WRTBV: +1,59% 17,20 17,21 17,29 16,93 17,21
ubuntu@linux-a-20:~/test$
```

- **List only those stocks which have positive change value (i.e. +xx,xx%) in the list**

```
ubuntu@linux-a-10:~$ cat stocks.txt | grep ': +'
Componenta Corporation :CTH1V: +2,06% 5,90 5,99 5,95 5,89 5,95
KCI Konecranes Plc :KCI1V: +0,06% 34,10 34,16 34,20 34,05 34,17
Outokumpu Oyj :OUT1V: +0,98% 13,36 13,37 13,42 13,27 13,36
Rautaruukki Corporation :RTRKS: +0,24% 8,46 8,50 8,52 8,42
Wärtsilä Corporation A :WRTAV: +3,47% 16,82 17,00 17,00 16,74 17,00
Wärtsilä Corporation B :WRTBV: +1,59% 17,20 17,21 17,29 16,93 17,21
Stora Enso Oyj A :STEAV: +1,31% 11,50 11,58 11,58 11,53 11,58
Stromsdal Corporation B :STMBS: +0,43% 2,00 2,06 0,00 0,00 2,10
```

Question 58: Get nimipaivat.txt (Finnish name days) textfile from here nimipaivat.txt

Answer 58:

- **From nimipaivat.txt, find out how many names start with a letter A and end to a letter i?**

```
ubuntu@linux-a-10:~$ cat nimipaivat.txt | grep '^A.*i '
```

- **How can you convert previous names to lower-case?**

```
ubuntu@linux-a-10:~$ cat nimipaivat.txt | grep '^A.*i ' | tr "A-Z" "a-z"
```

- **From previous names, who are celebrating in December?**

```
ubuntu@linux-a-10:~$ cat nimipaivat.txt | grep '^A.*i ' | tr "A-Z" "a-z" | grep '12'
airi 4.12.
anneli 9.12.
anni 9.12.
annikki 9.12.
auli 16.12.
aulikki 16.12.
aatami 24.12.
```

```
ubuntu@linux-a-10:~$ cat nimipaivat.txt | grep '^A.*i ' | tr "A-Z" "a-z" | grep '\.4\.$'
altti 24.4.
anselmi 21.4.
anssi 21.4.
ubuntu@linux-a-10:~$
```

- **From all names in nimipaivat.txt, search those who celebrate either 1st, 2nd or 3rd day in any month.**

```
ubuntu@linux-a-10:~$ cat nimipaivat.txt | grep ' [123]\..*\.$'
```

Question 59: Use lynx -dump "url" to print webpage to STDOUT

Answer 59:

- **Filter output so that you will get only the current Lotto numbers, but nothing else from the webpage**

  lynx -dump https://yle.fi/tekstitv/txt/471_0001.htm | grep 'OIKEAT NUMEROT' | cut -d':' -f2-

```
ubuntu@linux-a-20:~/test$ lynx -dump https://yle.fi/tekstitv/txt/471_0001.htm | grep 'OIKEAT NUMEROT' | cut -d':' -f2-
 6,11,26,27,31,35,37
```

Question 60: HTTP access to XML:

Answer 60:

- **Use Gnu tools or Cmder's Curl and Grep (and maybe other command line tools) to create a one-liner, which downloads the XML file and parses current temperature from FMI's weather station. One-liner must print only the current temperature in Oulu and nothing else. Command line one-liner and output should look something like this:**

  curl -s -L 'https://opendata.fmi.fi/wfs?request=getFeature&storedquery_id=fmi::observations::weather::timevaluepair&place=oulu&timestep=10&parameters=temperature' | tail -15 | head -1 | cut -c23-27

```
ubuntu@linux-a-20:~/test$ curl -s -L 'https://opendata.fmi.fi/wfs?request=getFeature&storedquery_id=fmi::observations::weather::timevaluepair&place=oulu&times
tep=10&parameters=temperature' | tail -15 | head -1 | cut -c23-27
-15.6
ubuntu@linux-a-20:~/test$
```

Question 61: Combine these two files to a single file with command line Gnu text tools

Answer 61:

- **The first file has timestamps and the second file has IP addresses**

```
ubuntu@linux-a-10:~$ paste firstfile.txt secondfile.txt
```

- **Use : as delimeter between columns. Output should look something like this:]**

```
ubuntu@linux-a-10:~$ paste firstfile.txt secondfile.txt -d":" > output.txt
```

Question 62: Delete unnecessary files created in this practice

Answer 62:done

Week 5

Question 63: Study this Telegram bot game version 0.1, this improved version 0.2 and even more improved version 0.3

Answer 63:

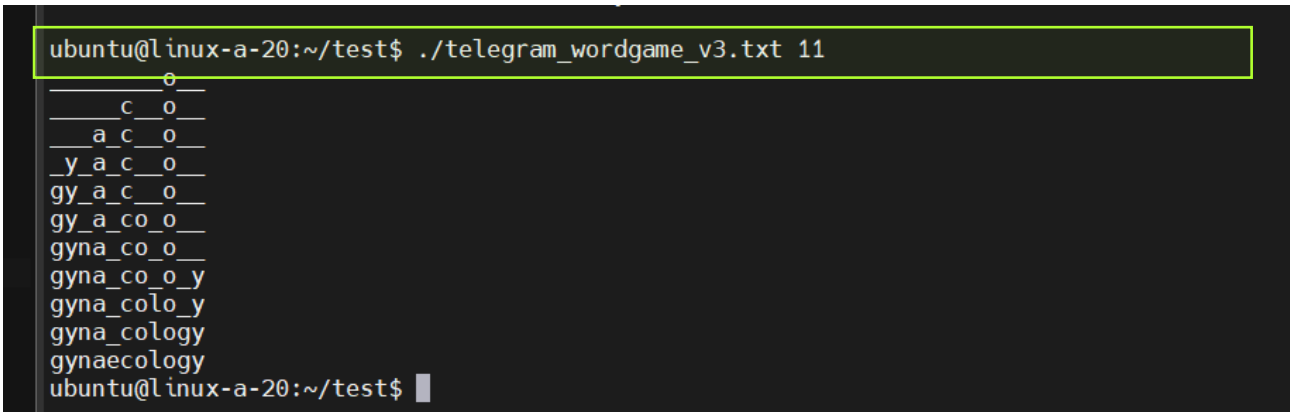
- **Compare all three scripts**

  v0.1, despite being longer and less parameterized, implements the revealing process.

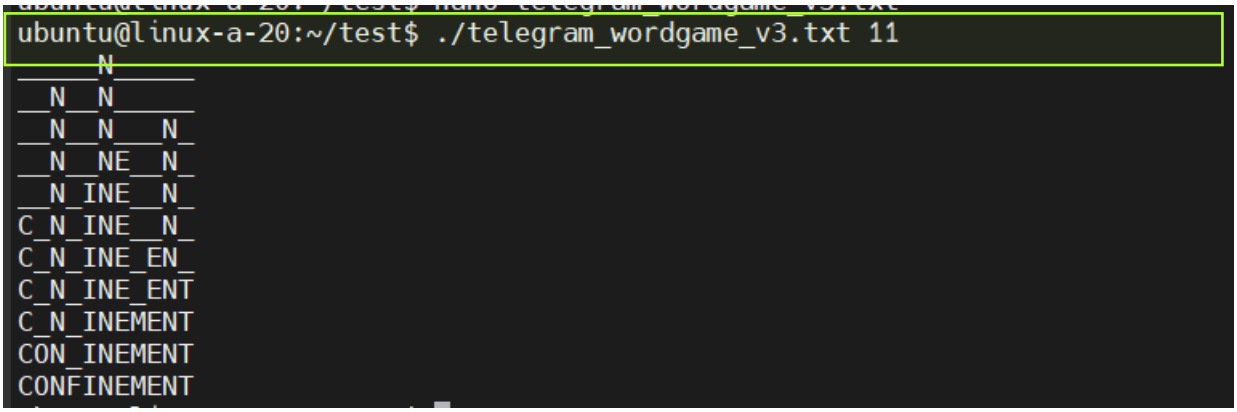
  v0.2 simplifies the script and enhances readability by using arrays for character reveal.

  v0.3, while slightly longer than v0.2, allows customization by setting the word length at the beginning.

- **Download the 3rd script (version 0.3) and related word list files (Github addresses are listed in the beginning of the script) with wget and run the script**



- **Modify the script to output all characters UPPER CASE**



```
#!/bin/bash

#
# v0.3 / TK
#


#
# Wordlists: https://github.com/verachell/English-word-lists-parts-of-speech-approximate
#
# https://raw.githubusercontent.com/verachell/English-word-lists-parts-of-speech-approximate/main/nouns/mostly-nouns.txt
# https://raw.githubusercontent.com/verachell/English-word-lists-parts-of-speech-approximate/main/other-categories/mostly-adjectives.txt
# https://raw.githubusercontent.com/verachell/English-word-lists-parts-of-speech-approximate/main/verbs/mostly-verbs-infinitive.txt
#

if [ $# -eq 0 ]; then
        echo "No word length supplied (4-21)"
        exit
fi

if [ $1 -lt 4 ] || [ $1 -gt 21 ]; then
        echo "Word length must be between 4 to 21 characters"
        exit
fi


word=($(cat mostly* | tr 'A-Z' 'a-z' | egrep "^[a-z]{$1}$" | shuf | head -1 | tr '[:lower:]' '[:upper:]' | fold -w1))
finalword=(_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _)

for i in $(seq 0 $(($1-1)) | shuf); do

        finalword[$i]=${word[$i]}
        reveal=$(echo ${finalword[@]} | sed 's/ //g' | cut -c1-$1)

        # data to telegram
        # curl -s -X POST "https://api.telegram.org/botNNNNNNNNNN:KEYYYYYY/sendMessage" -d "chat_id=XXXXXXXXX&text=Word puzzle with one minute delay: $reveal"
        # sleep 60

        # for debugging
        echo $reveal

done
                                                                [ Read 44 lines ]
```

- **Modify the script to add zero padded line numbers into the beginning of each line**

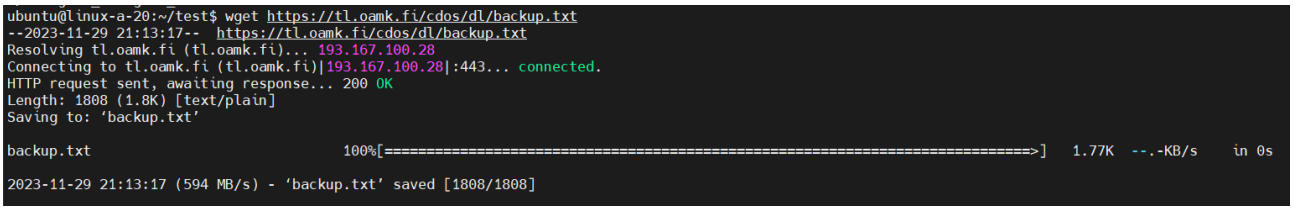
```
ubuntu@linux-a-20:~/test$ ./telegram_wordgame_v3.txt 11
01 _ _ _ _ _ _ C _ _ _ _
02 _ _ _ _ _ _ C _ _ N _
03 _ _ _ _ _ _ C _ _ N T
04 _ _ _ _ _ I C _ _ N T
05 _ _ E _ _ I C _ _ N T
06 _ _ E _ _ I C I _ N T
07 _ _ E F _ I C I _ N T
08 _ _ E F F I C I _ N T
09 C _ E F F I C I _ N T
10 C _ E F F I C I E N T
11 C O E F F I C I E N T
```

```
telegram_wordgam...  telegram_wordgame_...
1  #!/bin/bash
2
3  #
4  # v0.3 / TK
5  #
6
7  #
8  # Wordlists: https://github.com/verachell/English-word-lists-parts-of-speech-approximate
9  #
10 # https://raw.githubusercontent.com/verachell/English-word-lists-parts-of-speech-approximate/main/nouns/mostly-nouns.txt
11 # https://raw.githubusercontent.com/verachell/English-word-lists-parts-of-speech-approximate/main/other-categories/mostly-adjectives.txt
12 # https://raw.githubusercontent.com/verachell/English-word-lists-parts-of-speech-approximate/main/verbs/mostly-verbs-infinitive.txt
13 #
14
15 if [ $# -eq 0 ]; then
16   echo "No word length supplied (4-21)"
17   exit
18 fi
19
20 if [ $1 -lt 4 ] || [ $1 -gt 21 ]; then
21   echo "Word length must be between 4 to 21 characters"
22   exit
23 fi
24
25
26 word=($(cat mostly* | tr 'A-Z' 'a-z' | egrep "^[a-z]{$1}$" | shuf | head -1 | tr '[:lower:]' '[:upper:]' | fold -w1))
27 finalword=(_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _)
28 linenumber=1;
29 for i in $(seq 0 $(($1-1)) | shuf); do
30
31   finalword[$i]=${word[$i]}
32   reveal=$(echo ${finalword[@]} | sed 's/ //g' | cut -c1-$1)
33
34   # data to telegram
35   # curl -s -X POST "https://api.telegram.org/botNNNNNNNNNN:KEYYYYYY/sendMessage" -d "chat_id=XXXXXXXXX&text=Word puzzle with one minute delay: $reveal" > /dev/null
36   # sleep 60
37
38   # for debugging
39   paddednumber=$(printf "%02d\n" $linenumber)
40   echo "$paddednumber $reveal"
41   ((linenumber++))
42 done
43
44 done
45
46
47
48
```

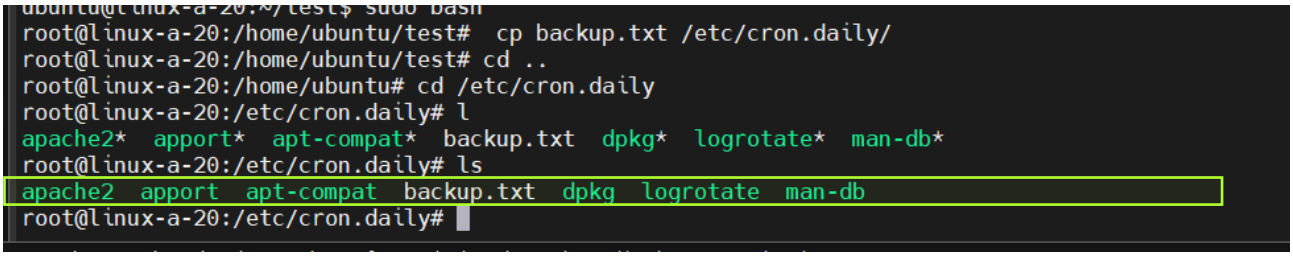
Question 64: Study and try [this simple incremental / full backup example script](). Do automatic backups as a root user:

Answer 64:

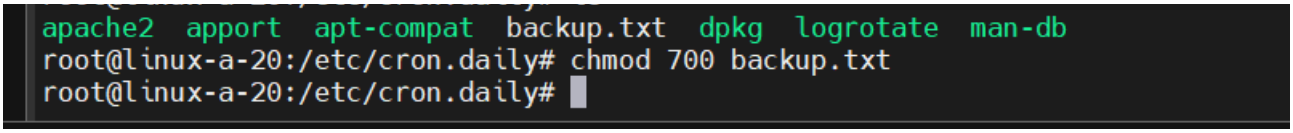
- **Use wget to download the backup script example**



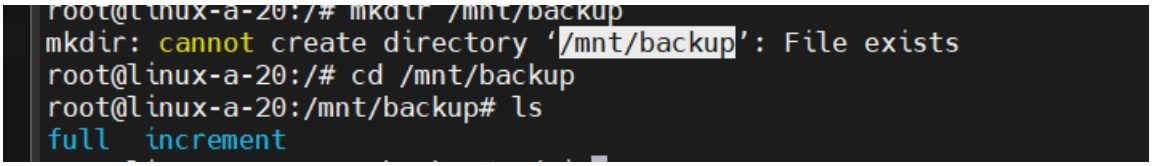
- **Move the downloaded file to /etc/cron.daily/ and rename it to backup (cron does not like if filename end to .bash. leave it out)**



- **Set permissions to 700 (and make sure root user is the file owner)**



- **Run the backup script from command line and check that it worked**



- **Uncomment initial delay lines from the script to create a random delay before the backup script executes all those heavy disk IO (find and tar) backup operations**

```
#!/bin/bash

# example of simple incremental / full backup script

# 1-6000 second random delay (prevents all student virtual hosts from taking backups simultaneously)
# uncomment these two lines before implementing to the server with crontab scheduler:

mydelay=$(shuf -i 1-6000 -n 1)
sleep $mydelay

# testing and creating backup directories
if [ ! -e /mnt/backup/ ]; then mkdir /mnt/backup; fi
if [ ! -e /mnt/backup/increment ]; then mkdir /mnt/backup/increment; fi
if [ ! -e /mnt/backup/full ]; then mkdir /mnt/backup/full; fi

# filesize set to unlimited and default file permissions
ulimit -f unlimited
umask 066

# setting locations of individual backups
DATA1="/etc"
DATA2="/usr/local"
DATA3="/home"
LIST=$(mktemp) # temporary filename for filename list(s)
set $(date) # date output will go to $1, $2 etc

# Full backups if Sunday
if test "$1" = "Sun" ; then
tar cfz "/mnt/backup/full/full_backup_etc.tar.gz" $DATA1 >& /dev/null
tar cfz "/mnt/backup/full/full_backup_usr_local.tar.gz" $DATA2 >& /dev/null
tar cfz "/mnt/backup/full/full_backup_home.tar.gz" $DATA3 >& /dev/null
```

- **Create some temporary test directory to /tmp and copy one of those smaller incremental backup files there**

```
root@linux-a-20:/mnt/backup/increment# ls
increment_etc_2023-Dec-7.tar.gz   increment_home_2023-Dec-7.tar.gz   increment_usr_local_2023-Dec-7.tar.gz
increment_etc_2023-Nov-29.tar.gz  increment_home_2023-Nov-29.tar.gz  increment_usr_local_2023-Nov-29.tar.gz
root@linux-a-20:/mnt/backup/increment# cp increment_etc_2023-Dec-7.tar.gz /tmp/test_backup/
cp: cannot create regular file '/tmp/test_backup/': Not a directory
root@linux-a-20:/mnt/backup/increment# cp increment_etc_2023-Dec-7.tar.gz /tmp/backup_test/
root@linux-a-20:/mnt/backup/increment# cp increment_home_2023-Dec-7.tar.gz /tmp/backup_test/
root@linux-a-20:/mnt/backup/increment# cd /tmp/backup_test/
root@linux-a-20:/tmp/backup_test# ls
increment_etc_2023-Dec-7.tar.gz  increment_home_2023-Dec-7.tar.gz
root@linux-a-20:/tmp/backup_test#
```

- **Use tar to list contents of the copied package**

```
root@linux-a-20:/tmp/backup_test# tar -tvf increment_etc_2023-Dec-7.tar.gz
-rwx------ ubuntu/ubuntu  1808 2022-12-04 17:57 etc/cron.daily/backup.txt
-rw-r--r-- root/root       152 2023-12-06 22:53 etc/apache2/sites-available/your_configuration_file.conf
-rw-r--r-- root/root      7223 2023-12-06 23:07 etc/apache2/apache2.conf
-rw-r--r-- root/root     23435 2023-12-07 06:36 etc/ld.so.cache
root@linux-a-20:/tmp/backup_test#
```

- **Use tar to decompress package and check that you managed to extract all the files from the package**

```
root@linux-a-20:/tmp/backup_test# tar -tvf increment_etc_2023-Dec-7.tar.gz
-rwx------ ubuntu/ubuntu  1808 2022-12-04 17:57 etc/cron.daily/backup.txt
-rw-r--r-- root/root       152 2023-12-06 22:53 etc/apache2/sites-available/your_configuration_file.conf
-rw-r--r-- root/root      7223 2023-12-06 23:07 etc/apache2/apache2.conf
-rw-r--r-- root/root     23435 2023-12-07 06:36 etc/ld.so.cache
root@linux-a-20:/tmp/backup_test# tar -xvf increment_etc_2023-Dec-7.tar.gz
etc/cron.daily/backup.txt
etc/apache2/sites-available/your_configuration_file.conf
etc/apache2/apache2.conf
etc/ld.so.cache
root@linux-a-20:/tmp/backup_test# ls
etc  increment_etc_2023-Dec-7.tar.gz  increment_home_2023-Dec-7.tar.gz
root@linux-a-20:/tmp/backup_test# cd
```


- **Remember that your virtual server host server is not taking any backups. It is up to you to backup and transfer your files if servers go down!**

- **Test how to pipe tar archived and compressed file through ssh connection to the students.oamk.fi server (students.oamk.fi accepts your student.oamk.fi credentials for SSH)**

```
root@linux-a-20:/tmp/backup_test# scp increment_etc_2023-Dec-7.tar.gz t3lixu00@s
tudents.oamk.fi:~/
t3lixu00@students.oamk.fi's password:
increment_etc_2023-Dec-7.tar.gz               100% 9077     2.0MB/s   00:00
root@linux-a-20:/tmp/backup_test#
```

Question 65: Create a file size comparison Bash script:

Answer 65:

- **It will prompt user to enter two filenames and then compares if the first given file is smaller, greater or equal size to the second file**

```
ubuntu@linux-a-20:~/test$ nano compare.bash
ubuntu@linux-a-20:~/test$ chmod 755 compare.bash
ubuntu@linux-a-20:~/test$ ./compare.bash
Enter name of first file:
/usr/bin/tail
Enter name of second file:
/usr/bin/tail
File /usr/bin/tail is the same size as /usr/bin/tail.
ubuntu@linux-a-20:~/test$ ./sizecomparison.bash
```

- **Before comparison, the script must check if given files exist**

```
ubuntu@linux-a-20:~/test$ ./compare.bash
Enter name of first file:
/asfdsa/das
Enter name of second file:
/kdfk/sad
Error: File '/asfdsa/das' does not exist.
ubuntu@linux-a-20:~/test$
port MobaXterm by subscribing to the professional edition here:  https://mobaxterm.mobatek.
```

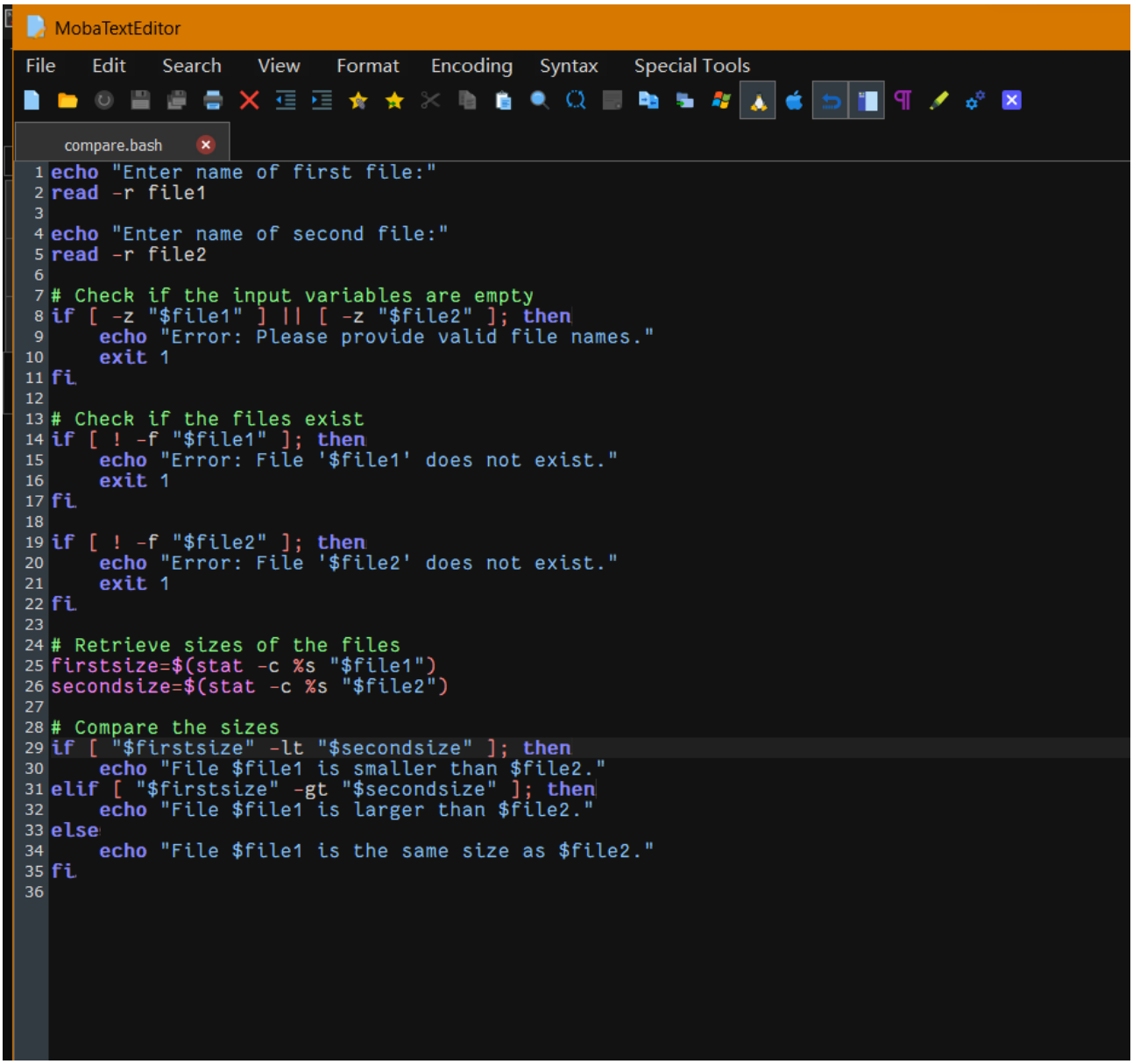
- **If the file does not exist, script will print an error message**

```
ubuntu@linux-a-20:~/test$ ./compare.bash
Enter name of first file:
/asfdsa/das
Enter name of second file:
/kdfk/sad
Error: File '/asfdsa/das' does not exist.
ubuntu@linux-a-20:~/test$
port MobaXterm by subscribing to the professional edition here:  https://mobaxterm.mobatek.
```

- **Use Bash if expressions to check that file exists and to test which file was larger**

```
ubuntu@linux-a-20:~/test$ nano compare.bash
ubuntu@linux-a-20:~/test$ chmod 755 compare.bash
ubuntu@linux-a-20:~/test$ ./compare.bash
Enter name of first file:
/usr/bin/tail
Enter name of second file:
/usr/bin/tail
File /usr/bin/tail is the same size as /usr/bin/tail.
ubuntu@linux-a-20:~/test$ ./sizecomparison.bash
```

- **Use "read" to ask filenames interactively**



Question 66: Change previous script to accept two filenames as a command line parameters. Example output should be something like this:

Answer 66:

```
ubuntu@linux-a-20:~/test$ ./compare2.bash
/usr/bin/tail /usr/bin/head
File /usr/bin/tail is larger
ubuntu@linux-a-20:~/test$
```

- **Also, script must print an error message and exit if it does not receive two parameters (the filenames) from command line**

```
  GNU nano 6.2
IFS=" "  read -r file1 file2
# Check if the input variables are empty
if [ -z "$file1" ] || [ -z "$file2" ]; then
    echo "Error: Please provide valid file names."
    exit 1
fi

# Check if the files exist
if [ ! -f "$file1" ]; then
    echo "Error: File '$file1' does not exist."
    exit 1
fi

if [ ! -f "$file2" ]; then
    echo "Error: File '$file2' does not exist."
    exit 1
fi

# Retrieve sizes of the files
firstsize=$(stat -c %s "$file1")
secondsize=$(stat -c %s "$file2")

# Compare the sizes
if [ "$firstsize" -lt "$secondsize" ]; then
    echo "File $file2 is larger."
elif [ "$firstsize" -gt "$secondsize" ]; then
    echo "File $file1 is larger"
else
    echo "File $file1 is the same size as $file2."
fi
```

Question 67: Think why second solution (to use parameters after the command instead of interactive input) is usually more practical solution?

Answer 67:

Because its faster and convineint

Question 68: Create “Rock Scissor Paper”-game with Bash

Answer 68:

- **Script will prompt user to pick either Rock, Scissors or Paper**

```
ubuntu@linux-a-20:~/test$ ./game2.bash
Welcome to Rock, Paper, Scissors!
Enter your choice (Rock/Paper/Scissors):
Rock
You chose: rock
Computer chose: Rock
It's a tie! Both chose Rock.
ubuntu@linux-a-20:~/test$
```

- **Script will randomise one option (computer player’s selection) and return results**

```
#!/bin/bash

# Function to get computer's choice
getComputerChoice() {
    choices=("Rock" "Paper" "Scissors")
    computer_choice=${choices[$((RANDOM % ${#choices[@]}))]}
}

# Function to determine the winner
determineWinner() {
    user_choice=$1
    case $user_choice in
        "Rock" | "rock")
            case $computer_choice in
                "Rock")
                    result="It's a tie! Both chose Rock."
                    ;;
                "Paper")
                    result="Computer wins! Paper covers Rock."
                    ;;
                "Scissors")
                    result="You win! Rock crushes Scissors."
                    ;;
            esac
            ;;
        "Paper" | "paper")
            case $computer_choice in
                "Rock")
                    result="You win! Paper covers Rock."
                    ;;
                "Paper")
                    result="It's a tie! Both chose Paper."
                    ;;
                "Scissors")
                    result="Computer wins! Scissors cuts Paper."
                    ;;
            esac
            ;;
        "Scissors" | "scissors")
            case $computer_choice in
                "Rock")
                    result="Computer wins! Rock crushes Scissors."
                    ;;
                "Paper")
                    result="You win! Scissors cuts Paper."
                    ;;
                "Scissors")
                    result="It's a tie! Both chose Scissors."
                    ;;
            esac
            ;;
        *)
            result="Invalid choice. Please enter Rock, Paper, or Scissors."
            ;;
    esac
}

# Function to display the results
displayResults() {
    echo "You chose: $user_choice"
    echo "Computer chose: $computer_choice"
    echo "$result"
}

# Main game logic
echo "Welcome to Rock, Paper, Scissors!"
echo "Enter your choice (Rock/Paper/Scissors):"
read user_choice

# Get computer's choice
getComputerChoice

# Convert input to lowercase for case-insensitive comparison
user_choice=$(echo "$user_choice" | tr '[:upper:]' '[:lower:]')

# Determine the winner
determineWinner "$user_choice"
```

- **Rules are: Rock wins scissors. Paper wins rock. Scissors wins paper**

```bash
#!/bin/bash

for name in $(cat nimipaivat.txt | cut -d" " -f1); do
    sqlite3 mynamedays.db "INSERT INTO namedata VALUES(\"$name\");"
done

#sqlite3 mynamedays.db "SELECT * FROM namedata;"
```

Question 69: Create a welcoming script which will check current time and will echo welcome message if time is:

Answer 69:

```
ubuntu@linux-a-20:~/test$ ./greeting.bash
Good night, Ubuntu!
```

```
  GNU nano 6.2                                                    greeting
#!/bin/bash

# Get the current hour
current_hour=$(date +%H)

# Get the username from /etc/passwd for the current user
username=$(id -un)

# Get the real name of the user from /etc/passwd
real_name=$(getent passwd "$username" | cut -d: -f5 | cut -d, -f1)

# Display a welcome message based on the current time
if [[ "$current_hour" -ge 6 && "$current_hour" -lt 12 ]]; then
    echo "Good morning, $real_name!"
elif [[ "$current_hour" -ge 12 && "$current_hour" -lt 17 ]]; then
    echo "Good day, $real_name!"
elif [[ "$current_hour" -ge 18 && "$current_hour" -lt 22 ]]; then
    echo "Good evening, $real_name!"
else
    echo "Good night, $real_name!"
fi
```

Question 70: Create a script which will check current date and searches all persons having finnish name day at the moment. You can use nimipaivat.txt as a datafile

Answer 70:

```
ubuntu@linux-a-20:~/test$ ./nameday.bash
$ date:
Thu Dec 07 19:46:41 EET 2023

$ ./nameday.bash
Sampsa 7.12.
ubuntu@linux-a-20:~/test$
```

```
current_time=$(date +"%a %b %d %T %Z %Y")
echo -e "$ date:\n$current_time"
time=$(date +"%d-%m" |  sed 's/0//g; s/-/./g')
stime=" $time"
findname=$(grep "$stime" nimipaivat.txt)
echo -e "\n$ ./nameday.bash\n$findname"
```

Question 71: Install SQLite3 database engine and tools to the Linux server (basic tutorial here) and:

Answer 71:

- **Download nimipaivat.txt text file**

```
ubuntu@linux-a-20:~/test$ wget https://tl.oamk.fi/cdos/dl/nimipaivat.txt
--2023-11-30 13:29:19--  https://tl.oamk.fi/cdos/dl/nimipaivat.txt
Resolving tl.oamk.fi (tl.oamk.fi)... 193.167.100.28
Connecting to tl.oamk.fi (tl.oamk.fi)|193.167.100.28|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 9636 (9.4K) [text/plain]
Saving to: 'nimipaivat.txt'

nimipaivat.txt                      100%[=============================================================================>]   9.41K  --.-KB/s    in 0s

2023-11-30 13:29:19 (55.5 MB/s) - 'nimipaivat.txt' saved [9636/9636]

ubuntu@linux-a-20:~/test$ ls
backup.txt  compare.bash  greeting.bash  nimipaivat.txt
```

- **Create a new SQLite database**

```
ubuntu@linux-a-20:~/test$ sqlite3 mynamedays.db
SQLite version 3.37.2 2022-01-06 13:25:41
Enter ".help" for usage hints.
sqlite> ls
   ...> CREATE TABLE namedata(name text NOT NULL);
Error: in prepare, near "ls": syntax error (1)
sqlite> CREATE TABLE namedata(name text NOT NULL);
sqlite> INSERT INTO namedata VALUES("Test data");
sqlite> SELECT * FROM namedata;
Test data
sqlite>
```

- **Create a Bash script (or use Bash one-liner) which inserts only first names but not dates from the nimipaivat.txt to the SQLite database as a single column data**

```
#!/bin/bash

for name in $(cat nimipaivat.txt | cut -d" " -f1); do
    sqlite3 mynamedays.db "INSERT INTO namedata VALUES(\"$name\");"
done

#sqlite3 mynamedays.db "SELECT * FROM namedata;"
```

- **Create a Bash script which will fetch one single random name from the SQLite database**

```
#!/bin/bash

sqlite3 mynamedays.db "SELECT * FROM namedata;" |  shuf | head -1
```

```
#!/bin/bash

for name in $(cat nimipaivat.txt | cut -d" " -f1); do
    sqlite3 mynamedays.db "INSERT INTO namedata VALUES(\"$name\");"
done

#sqlite3 mynamedays.db "SELECT * FROM namedata;"
```

Week 6

Question 72: Install Apache web server to your Linux server if it isn't installed already

Answer 72:

- **Ignore/skip possible ufw (firewall) parts. Ufw has been disabled intentionally on student virtual servers**

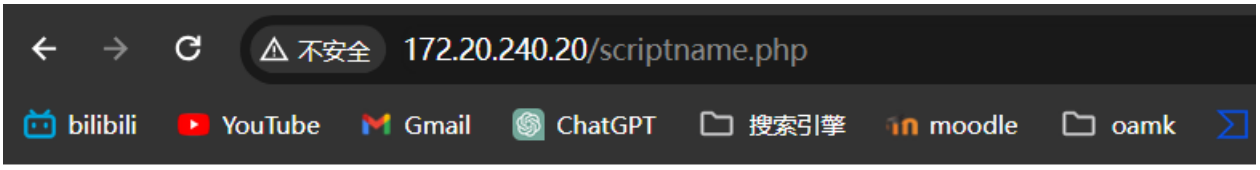
  systemctl status apache2

```
Last login: Sun Dec  3 18:29:50 2023 from 10.2.122.84
ubuntu@linux-a-20:~$ systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor prese>
     Active: active (running) since Thu 2023-11-23 06:35:55 EET; 1 week 5 days >
       Docs: https://httpd.apache.org/docs/2.4/
    Process: 70370 ExecReload=/usr/sbin/apachectl graceful (code=exited, status>
   Main PID: 45680 (apache2)
      Tasks: 55 (limit: 2309)
     Memory: 5.2M
        CPU: 1min 6.739s
     CGroup: /system.slice/apache2.service
             ├─45680 /usr/sbin/apache2 -k start
             ├─70374 /usr/sbin/apache2 -k start
             └─70375 /usr/sbin/apache2 -k start

Dec 02 00:00:17 linux-a-20 systemd[1]: Reloaded The Apache HTTP Server.
Dec 03 00:00:18 linux-a-20 systemd[1]: Reloading The Apache HTTP Server...
Dec 03 00:00:18 linux-a-20 apachectl[67616]: AH00558: apache2: Could not reliab>
Dec 03 00:00:18 linux-a-20 systemd[1]: Reloaded The Apache HTTP Server.
Dec 04 00:00:41 linux-a-20 systemd[1]: Reloading The Apache HTTP Server...
Dec 04 00:00:41 linux-a-20 apachectl[69373]: AH00558: apache2: Could not reliab>
Dec 04 00:00:41 linux-a-20 systemd[1]: Reloaded The Apache HTTP Server.
Dec 05 00:00:17 linux-a-20 systemd[1]: Reloading The Apache HTTP Server...
Dec 05 00:00:18 linux-a-20 apachectl[70373]: AH00558: apache2: Could not reliab>
Dec 05 00:00:18 linux-a-20 systemd[1]: Reloaded The Apache HTTP Server.
lines 1-24...skipping...
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
     Active: active (running) since Thu 2023-11-23 06:35:55 EET; 1 week 5 days ago
       Docs: https://httpd.apache.org/docs/2.4/
    Process: 70370 ExecReload=/usr/sbin/apachectl graceful (code=exited, status=0/SUCCESS)
   Main PID: 45680 (apache2)
      Tasks: 55 (limit: 2309)
     Memory: 5.2M
        CPU: 1min 6.739s
     CGroup: /system.slice/apache2.service
             ├─45680 /usr/sbin/apache2 -k start
             ├─70374 /usr/sbin/apache2 -k start
```

- **Add PHP support to your Apache web server. See the MySQL example with PHP for required software packages**

```
ubuntu@linux-a-20:~$ sudo bash
apt install apache2 php php-cli libapache2-mod-php php-mysql mysql-server
root@linux-a-20:/home/ubuntu# apt install apache2 php php-cli libapache2-mod-php php-mysql mysql-server
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
apache2 is already the newest version (2.4.52-1ubuntu4.7).
The following additional packages will be installed:
  libapache2-mod-php8.1 libcgi-fast-perl libcgi-pm-perl libclone-perl libencode-locale-perl libevent-pthreads-2.1-7 libfcgi-bin libfcgi-perl libfcgi0ldbl
  libhtml-parser-perl libhtml-tagset-perl libhtml-template-perl libhttp-date-perl libhttp-message-perl libio-html-perl liblwp-mediatypes-perl libmecab2
  libprotobuf-lite23 libtimedate-perl liburi-perl mecab-ipadic mecab-ipadic-utf8 mecab-utils mysql-client-8.0 mysql-client-core-8.0 mysql-common
  mysql-server-8.0 mysql-server-core-8.0 php-common php8.1 php8.1-cli php8.1-common php8.1-mysql php8.1-opcache php8.1-readline
Suggested packages:
  php-pear libdata-dump-perl libipc-sharedcache-perl libbusiness-isbn-perl libwww-perl mailx tinyca
The following NEW packages will be installed:
  libapache2-mod-php libapache2-mod-php8.1 libcgi-fast-perl libcgi-pm-perl libclone-perl libencode-locale-perl libevent-pthreads-2.1-7 libfcgi-bin
  libfcgi-perl libfcgi0ldbl libhtml-parser-perl libhtml-tagset-perl libhtml-template-perl libhttp-date-perl libhttp-message-perl libio-html-perl
  liblwp-mediatypes-perl libmecab2 libprotobuf-lite23 libtimedate-perl liburi-perl mecab-ipadic mecab-ipadic-utf8 mecab-utils mysql-client-8.0
  mysql-client-core-8.0 mysql-common mysql-server mysql-server-8.0 mysql-server-core-8.0 php php-cli php-common php-mysql php8.1 php8.1-cli php8.1-common
  php8.1-mysql php8.1-opcache php8.1-readline
0 upgraded, 40 newly installed, 0 to remove and 23 not upgraded.
Need to get 34.8 MB of archives.
After this operation, 265 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://archive.ubuntu.com/ubuntu jammy/main amd64 mysql-common all 5.8+1.0.8 [7212 B]
Get:2 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 mysql-client-core-8.0 amd64 8.0.35-0ubuntu0.22.04.1 [2682 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 mysql-client-8.0 amd64 8.0.35-0ubuntu0.22.04.1 [22.7 kB]
Get:4 http://archive.ubuntu.com/ubuntu jammy/main amd64 libevent-pthreads-2.1-7 amd64 2.1.12-stable-1build3 [7642 B]
Get:5 http://archive.ubuntu.com/ubuntu jammy/main amd64 libmecab2 amd64 0.996-14build9 [199 kB]
Get:6 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libprotobuf-lite23 amd64 3.12.4-1ubuntu7.22.04.1 [209 kB]
Get:7 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 mysql-server-core-8.0 amd64 8.0.35-0ubuntu0.22.04.1 [17.6 MB]
Get:8 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 mysql-server-8.0 amd64 8.0.35-0ubuntu0.22.04.1 [1438 kB]
```

- **Create following PHP script under the web server document root (/var/www/html) and test that your server is executing the PHP script when requesting it with a web browser (address is: http://server/scriptname.php):**



Tuesday
Tuesday 5th of December 2023 05:23:05 PM
Tue, 05 Dec 2023 17:23:05 +0000

```
  GNU nano 6.2
<?php
date_default_timezone_set('UTC');
echo date("l");
echo ("<br>");
echo date('l jS \of F Y h:i:s A');
echo ("<br>");
echo date(DATE_RFC2822);
?>
```

Question 73: Modify the Apache web server configuration to redirect HTTP GET requests to the directory /weather/ to https://wttr.in/. You will most likely need to enable the redirect module for Apache

Answer 73:

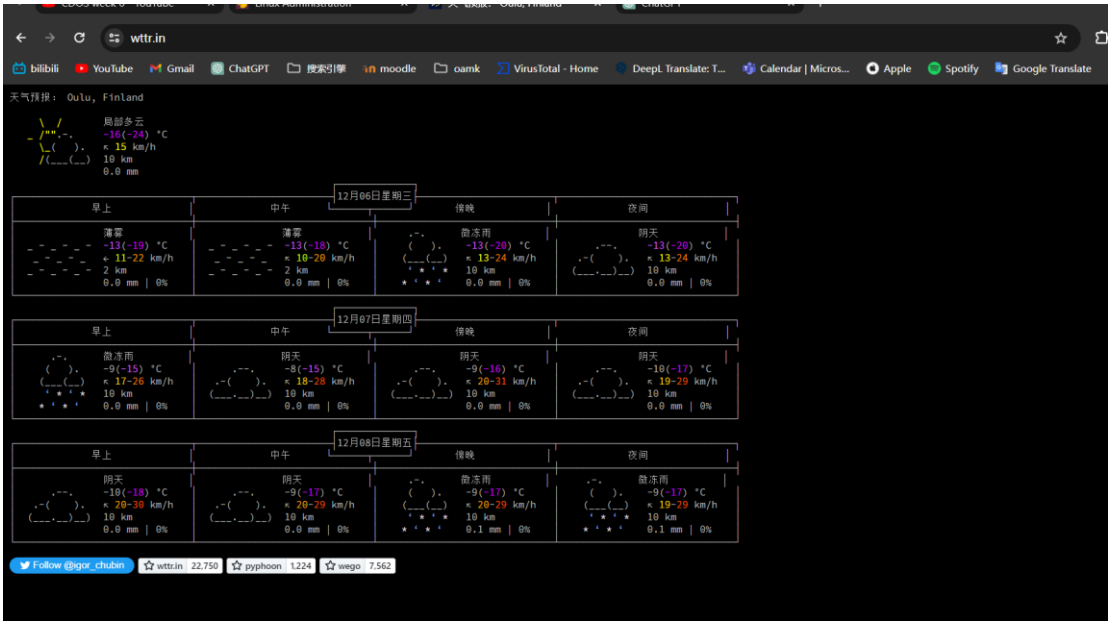
- **So, visiting your server http://IP_or_DNSname/weather/ takes the browser to wttr.in. Check with web browser that the redirect works as intended**

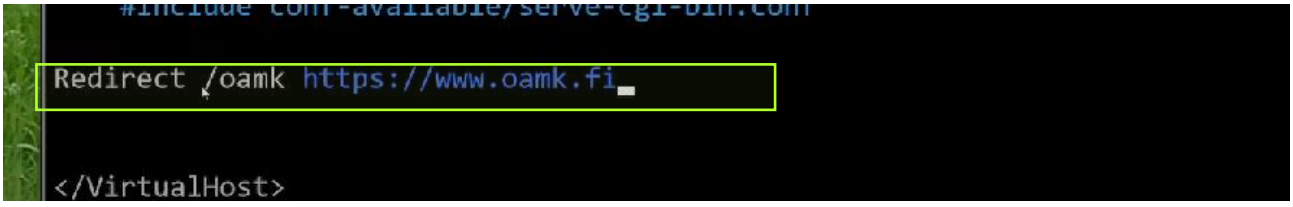
  Way1

```
root@linux-a-20:/var/www/html# cd /etc/apache2/
root@linux-a-20:/etc/apache2# ls
apache2.conf  conf-available  conf-enabled  envvars  magic  mods-available  mods-enabled  ports.conf  sites-available  sites-enabled
root@linux-a-20:/etc/apache2# nano apache2.conf
root@linux-a-20:/etc/apache2# nano apache2.conf
root@linux-a-20:/etc/apache2# systemctl restart apache2
root@linux-a-20:/etc/apache2# cd /var/www/
root@linux-a-20:/var/www# .htaccess
.htaccess: command not found
root@linux-a-20:/var/www# sudo nano /var/www/html/weather/.htaccess
root@linux-a-20:/var/www#
```

```
# access here, or in any related virtual host.
<Directory />
        Options FollowSymLinks
        AllowOverride None
        Require all denied
</Directory>

<Directory /usr/share>
        AllowOverride None
        Require all granted
</Directory>

<Directory /var/www/>
        Options Indexes FollowSymLinks
        AllowOverride All
        Require all granted
</Directory>

#<Directory /srv/>
#       Options Indexes FollowSymLinks
#       AllowOverride None
#       Require all granted
#</Directory>



# AccessFileName: The name of the file to look for in each directory
# for additional configuration directives.  See also the AllowOverride
# directive.
```

Way2





Question 74: Check last entries in Apache access and error log files in /var/log/apache2/

Answer 74:

Access logs

tail -n 10 /var/log/apache2/access.log

```
access.log.2.gz  access.log.5.gz  error.log        error.log.11.gz  error.log.14.gz  error.log.4.gz  error.log.7.gz  other_vhosts_access.log
root@linux-a-20:/var/log/apache2# tail -n 10 /var/log/apache2/access.log
10.2.122.84 - - [05/Dec/2023:19:23:05 +0200] "GET /scriptname.php HTTP/1.1" 200 334 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/119.0.0.0 Safari/537.36"
10.2.122.84 - - [05/Dec/2023:19:23:05 +0200] "GET /favicon.ico HTTP/1.1" 404 491 "http://172.20.240.20/scriptname.php" "Mozilla/5.0 (Windows NT 10.0; Win64; x
64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.0.0 Safari/537.36"
10.2.122.84 - - [05/Dec/2023:19:23:30 +0200] "GET /measurement.php HTTP/1.1" 500 185 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
 like Gecko) Chrome/119.0.0.0 Safari/537.36"
10.2.122.84 - - [05/Dec/2023:19:23:32 +0200] "GET /measurement.php HTTP/1.1" 500 185 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
 like Gecko) Chrome/119.0.0.0 Safari/537.36"
10.2.122.84 - - [05/Dec/2023:19:23:32 +0200] "GET /measurement.php HTTP/1.1" 500 185 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
 like Gecko) Chrome/119.0.0.0 Safari/537.36"
10.2.122.84 - - [05/Dec/2023:19:23:33 +0200] "GET /measurement.php HTTP/1.1" 500 185 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
 like Gecko) Chrome/119.0.0.0 Safari/537.36"
10.2.122.84 - - [05/Dec/2023:19:23:33 +0200] "GET /measurement.php HTTP/1.1" 500 185 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
 like Gecko) Chrome/119.0.0.0 Safari/537.36"
10.2.122.84 - - [05/Dec/2023:19:23:33 +0200] "GET /measurement.php HTTP/1.1" 500 185 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
 like Gecko) Chrome/119.0.0.0 Safari/537.36"
10.2.122.84 - - [05/Dec/2023:19:24:25 +0200] "-" 408 0 "-" "-"
root@linux-a-20:/var/log/apache2#
```

Error logs

tail -n 10 /var/log/apache2/error.log

```
root@linux-a-20:/var/log/apache2# tail -n 10 /var/log/apache2/error.log
[Tue Dec 05 16:10:42.201662 2023] [core:notice] [pid 78944] AH00094: Command line: '/usr/sbin/apache2'
[Tue Dec 05 16:11:54.321127 2023] [mpm_prefork:notice] [pid 78944] AH00170: caught SIGWINCH, shutting down gracefully
[Tue Dec 05 16:11:54.471526 2023] [mpm_prefork:notice] [pid 79425] AH00163: Apache/2.4.52 (Ubuntu) configured -- resuming normal operations
[Tue Dec 05 16:11:54.471663 2023] [core:notice] [pid 79425] AH00094: Command line: '/usr/sbin/apache2'
[Tue Dec 05 19:23:30.947935 2023] [php:error] [pid 79428] [client 10.2.122.84:3142] PHP Fatal error:  Uncaught mysqli_sql_exception: Table 'measurements.rawda
ta' doesn't exist in /var/www/html/measurement.php:14\nStack trace:\n#0 /var/www/html/measurement.php(14): mysqli->query()\n#1 {main}\n  thrown in /var/www/ht
ml/measurement.php on line 14
[Tue Dec 05 19:23:32.662561 2023] [php:error] [pid 79427] [client 10.2.122.84:3144] PHP Fatal error:  Uncaught mysqli_sql_exception: Table 'measurements.rawda
ta' doesn't exist in /var/www/html/measurement.php:14\nStack trace:\n#0 /var/www/html/measurement.php(14): mysqli->query()\n#1 {main}\n  thrown in /var/www/ht
ml/measurement.php on line 14
[Tue Dec 05 19:23:32.871606 2023] [php:error] [pid 79430] [client 10.2.122.84:3145] PHP Fatal error:  Uncaught mysqli_sql_exception: Table 'measurements.rawda
ta' doesn't exist in /var/www/html/measurement.php:14\nStack trace:\n#0 /var/www/html/measurement.php(14): mysqli->query()\n#1 {main}\n  thrown in /var/www/ht
ml/measurement.php on line 14
[Tue Dec 05 19:23:33.058490 2023] [php:error] [pid 79429] [client 10.2.122.84:3146] PHP Fatal error:  Uncaught mysqli_sql_exception: Table 'measurements.rawda
ta' doesn't exist in /var/www/html/measurement.php:14\nStack trace:\n#0 /var/www/html/measurement.php(14): mysqli->query()\n#1 {main}\n  thrown in /var/www/ht
ml/measurement.php on line 14
[Tue Dec 05 19:23:33.478731 2023] [php:error] [pid 79867] [client 10.2.122.84:3147] PHP Fatal error:  Uncaught mysqli_sql_exception: Table 'measurements.rawda
ta' doesn't exist in /var/www/html/measurement.php:14\nStack trace:\n#0 /var/www/html/measurement.php(14): mysqli->query()\n#1 {main}\n  thrown in /var/www/ht
ml/measurement.php on line 14
[Tue Dec 05 19:23:33.658620 2023] [php:error] [pid 79426] [client 10.2.122.84:3148] PHP Fatal error:  Uncaught mysqli_sql_exception: Table 'measurements.rawda
ta' doesn't exist in /var/www/html/measurement.php:14\nStack trace:\n#0 /var/www/html/measurement.php(14): mysqli->query()\n#1 {main}\n  thrown in /var/www/ht
ml/measurement.php on line 14
root@linux-a-20:/var/log/apache2#
```

Question 75: Install MySQL server and create some basic database there with one or more tables and insert some data into the table(s). See the basic MySQL example with PHP

Answer 75:

- **Note: Do not use that mysql_secure_installation command I used in the lecture recording. mysql_secure_installation command is now obsolete. You can access MySQL root shell with Linux root user privileged like this:**

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| measurements       |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.02 sec)

mysql>
```

Question 76: Create a Bash script which will fetch and process data from marine traffic API

Answer 76:

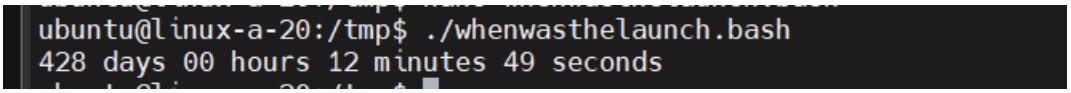
- **Script must download the JSON-file from marine traffic port call API and print how many ships are currently there? (Search vesselName from the JSON)**

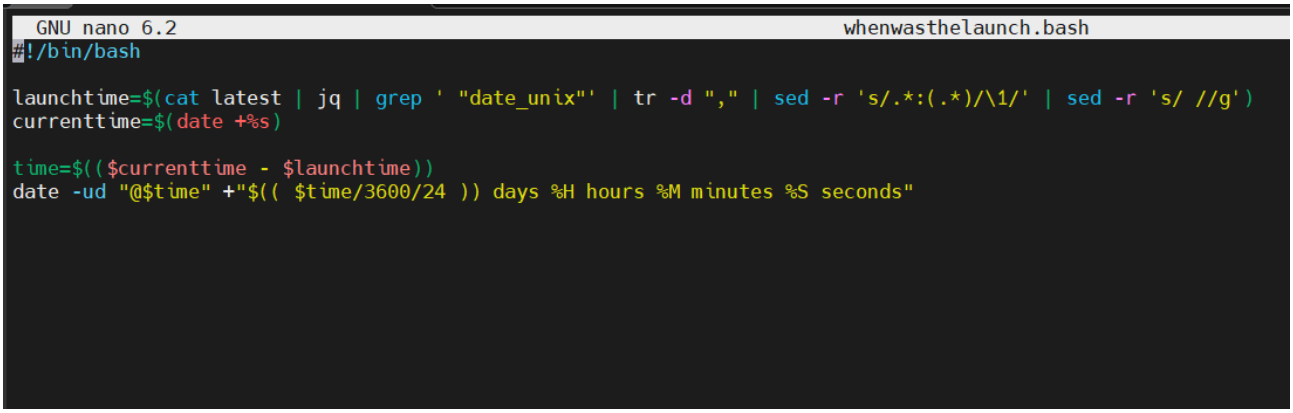
- **Filter the /tmp/result.json file data with jq or with GNU text utilities such as sed, awk, cut, grep etc. to only show how many ships are currently there? (Search vesselName from the JSON)**

Question 77: Create a Bash script which checks spaceX launch schedule API and tells how many days ago was the last launch. Result should be something like this:

Answer 77:

- **Tip: Use unix epoch timestamp from the API reply and current Unix epoch time from date command to calculate and show the time difference. date +%s outputs the current unix epoch time**

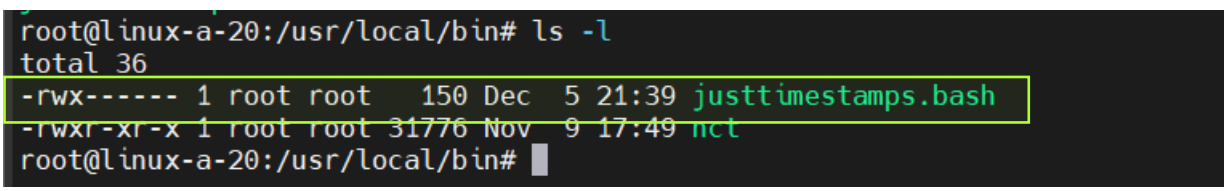




Question 78: Create this Bash script to /usr/local/bin directory. Name it to justtimestamps.bash and set the file permissions to 700 (root user has full access, others none). Check that the root is the file owner:

Answer 78:

- **Create a new systemd service configuration file timestampdemo.service to the /etc/systemd/system directory with contents:**



- **Manage the service:**

```
root@linux-a-20:/etc/systemd/system# nano timestampdemo.service
root@linux-a-20:/etc/systemd/system# nano iscsi.service
root@linux-a-20:/etc/systemd/system# ls
cloud-final.service.wants               iscsi.service                  rescue.target.wants        snap.lxd.daemon.service          sshd.service
cloud-init.target.wants                 mdmonitor.service.wants        sleep.target.wants         snap.lxd.daemon.unix.socket      sudo.service
dbus-org.freedesktop.resolve1.service   multi-user.target.wants        snap-core20-2015.mount     snap.lxd.user-daemon.service     sysinit.target.wants
dbus-org.freedesktop.timesync1.service  multipath-tools.service        snap-lxd-24322.mount       snap.lxd.user-daemon.unix.socket syslog.service
emergency.target.wants                  network-online.target.wants    snap-snapd-20092.mount     snapd.mounts.target.wants        timers.target.wants
final.target.wants                      open-vm-tools.service.requires snap-snapd-20290.mount     sockets.target.wants             timestampdemo.service
getty.target.wants                      paths.target.wants             snap.lxd.activate.service  sshd-keygen@.service.d           vmtoolsd.service
root@linux-a-20:/etc/systemd/system# grep time
```

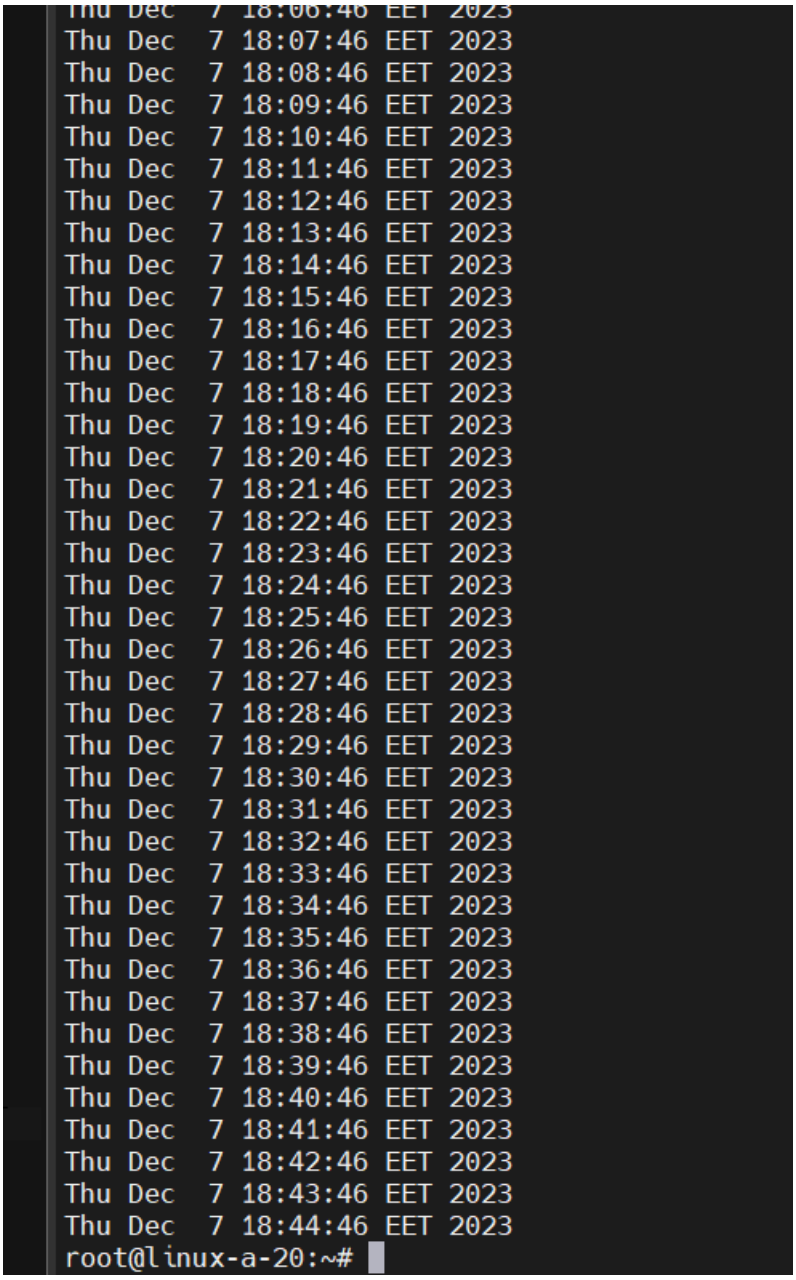
- **Reload service files with: systemctl daemon-reload**

```
root@linux-a-20:~# systemctl daemon-reload
root@linux-a-20:~#
```

- **Start the service with: systemctl start timestampdemo**

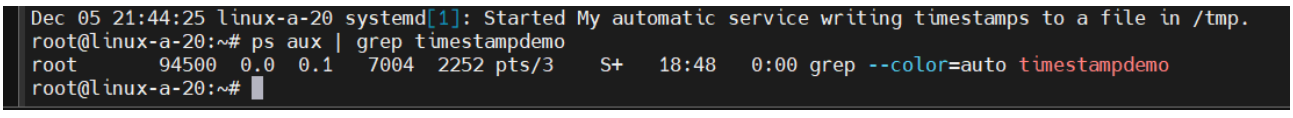
```
Thu Dec  7 18:44:46 EET 2023
root@linux-a-20:~# systemctl daemon-reload
root@linux-a-20:~# systemctl start timestampdemo
root@linux-a-20:~#
```

- **Check that /root/timestamp.txt file was created with the timestamp content (that date command should be running once per minute)**



- **Check the process list to verify that the script is running**

  ps aux | grep timestampdemo



- **Use command: systemctl status timestampdemo to check the service status**

```
  94494 pts/3    00:00:00 ps
root@linux-a-20:~# systemctl status timestampdemo
● timestampdemo.service - My automatic service writing timestamps to a file in /tmp
     Loaded: loaded (/etc/systemd/system/timestampdemo.service; disabled; vendor preset: enabled)
     Active: active (running) since Tue 2023-12-05 21:44:25 EET; 1 day 21h ago
   Main PID: 80314 (justtimestamps.)
      Tasks: 2 (limit: 2309)
     Memory: 792.0K
        CPU: 22.081s
     CGroup: /system.slice/timestampdemo.service
             ├─80314 /bin/bash /usr/local/bin/justtimestamps.bash
             └─94496 sleep 60

Dec 05 21:44:25 linux-a-20 systemd[1]: Started My automatic service writing timestamps to a file in /tmp.
root@linux-a-20:~#
```

- **Stop the script with: systemctl stop timestampdemo**

```
root@linux-a-20:~# systemctl stop timestampdemo
root@linux-a-20:~#
```

- **Verify from process list that service is not running anymore**

```
root@linux-a-20:~# ps aux | grep timestampdemo
root       94513  0.0  0.1   7004  2200 pts/3    S+   18:50   0:00 grep --color=auto timestampdemo
```

- **Use systemctl status timestampdemo to check the service status**

```
root@linux-a-20:~# systemctl status timestampdemo
○ timestampdemo.service - My automatic service writing timestamps to a file in /tmp
     Loaded: loaded (/etc/systemd/system/timestampdemo.service; disabled; vendor preset: enabled)
     Active: inactive (dead)

Dec 05 21:44:25 linux-a-20 systemd[1]: Started My automatic service writing timestamps to a file in /tmp.
Dec 07 18:49:43 linux-a-20 systemd[1]: Stopping My automatic service writing timestamps to a file in /tmp...
Dec 07 18:49:43 linux-a-20 systemd[1]: timestampdemo.service: Deactivated successfully.
Dec 07 18:49:43 linux-a-20 systemd[1]: Stopped My automatic service writing timestamps to a file in /tmp.
Dec 07 18:49:43 linux-a-20 systemd[1]: timestampdemo.service: Consumed 22.088s CPU time.
root@linux-a-20:~#
port MobaXterm by subscribing to the professional edition here:  https://mobaxterm.mobatek.net
```

- **Try running systemctl enable timestampdemo and systemctl disable timestampdemo (enables/disables the service during the server startup)**

```
root@linux-a-20:~# systemctl disable timestampdemo
root@linux-a-20:~# ^C
root@linux-a-20:~#  systemctl status timestampdemo
○ timestampdemo.service - My automatic service writing timestamps to a file in /tmp
     Loaded: loaded (/etc/systemd/system/timestampdemo.service; disabled; vendor preset: enabled)
     Active: inactive (dead)

Dec 05 21:44:25 linux-a-20 systemd[1]: Started My automatic service writing timestamps to a file in /tmp.
Dec 07 18:49:43 linux-a-20 systemd[1]: Stopping My automatic service writing timestamps to a file in /tmp...
Dec 07 18:49:43 linux-a-20 systemd[1]: timestampdemo.service: Deactivated successfully.
Dec 07 18:49:43 linux-a-20 systemd[1]: Stopped My automatic service writing timestamps to a file in /tmp.
Dec 07 18:49:43 linux-a-20 systemd[1]: timestampdemo.service: Consumed 22.088s CPU time.
root@linux-a-20:~#
port MobaXterm by subscribing to the professional edition here:  https://mobaxterm.mobatek.net
```

Week 7

Question 79: What is a Linux container? (with any technology such as LXC/LXD, Docker, Podman). Generic description what is a Linux container is enough.

Answer 79:

A Linux® container refers to a collection of one or more processes that operate in isolation from the system's remaining components. These containers rely on a distinct image that contains all the essential files needed to execute them. This design ensures the portability and consistency of Linux containers as they transition through various stages from development to testing and production. Their efficiency lies in their ability to expedite processes compared to conventional development pipelines that necessitate replicating traditional testing environments. The widespread adoption and user-friendly nature of containers also contribute significantly to their role in enhancing IT security. For instance, containers promote enhanced security practices by encapsulating applications and their dependencies, thereby reducing potential vulnerabilities.

Question 80: What is the difference between chroot, containers and full operating system virtualization (like Virtualbox, VMware, Hyper-V etc.)?

Answer 80:

Virtual machines (VMs) and containers stand as distinct technologies designed for running applications, each possessing unique attributes and applicability. VMs: VMs function as software emulations of physical computers, enabling the operation of multiple operating systems and applications on a single physical server. Each VM operates with its own isolated operating system, ensuring full hardware virtualization, including a dedicated kernel, memory, and resources. Their operation requires a hypervisor, a software layer responsible for creating, managing, and executing VMs. VMs ensure a high degree of isolation among applications, operating independently of each other. Due to encompassing an entire operating system, VMs possess a larger footprint and demand more resources than containers. VMs typically exhibit longer start-up and shutdown times in comparison to containers. Best suited for diverse operating system requirements, legacy applications, and scenarios necessitating robust application isolation. Containers: Containers facilitate operating system-level virtualization, allowing multiple segregated environments (containers) to operate on a single host OS. Containers share the host OS's kernel, libraries, and resources, leading to their lightweight and resource-efficient nature. Containerization platforms like Docker or Kubernetes manage container creation, deployment, and scaling. Containers boast rapid start-up and stoppage times, bypassing the need for booting an entire operating system. While offering less isolation than VMs by sharing the host OS, containerization technologies provide security features to mitigate potential risks. Highly portable, enabling easy packaging of applications and dependencies into a singular, self-contained unit. Ideal for deploying microservices, modernizing applications, and achieving swift scalability and deployment.

Question 81: What is Kubernetes?

Answer 81:

Kubernetes is an open-source tool used to manage and control groups of containers. It helps in automating

the processes involved in deploying and handling applications within these containers. Its main purpose is to simplify the management of complex applications made up of smaller, interconnected parts.

For instance, think of Kubernetes as a conductor leading an orchestra. The conductor (Kubernetes) ensures that each musician (container) plays their part at the right time, maintaining harmony within the music piece (application). It oversees how these containers are deployed, how they communicate, and manages their workload efficiently.

Question 82: What is Ansible?

Answer 82:

Ansible stands out as a widely respected open-source platform created using the Python programming language. This powerful tool acts as a facilitator for automating tasks, managing configurations, and deploying applications.

Imagine Ansible as a virtual assistant that helps streamline repetitive tasks for system administrators or developers. Just like a skilled assistant, it follows predefined instructions to handle various chores, ensuring consistency and efficiency in managing configurations and deploying applications across different systems.

For instance, think of Ansible as a conductor in an orchestra, guiding and coordinating different musicians (systems) to play their parts in harmony. It ensures that each system performs its designated tasks according to a set plan, orchestrating the overall performance smoothly.

Question83: What is Terraform?

Answer83:

HashiCorp Terraform serves as an infrastructure as code (IaC) tool, enabling the definition of cloud and on-premises resources through human-readable configuration files. These configurations can be versioned, reused, and easily shared. Using a unified approach, Terraform offers a consistent workflow to provision and oversee the entire lifecycle of your infrastructure.

This tool is versatile, capable of managing various tiers of components. It handles low-level entities such as compute, storage, and networking resources, while also managing high-level components like DNS entries and Software as a Service (SaaS) features. By employing Terraform, users can efficiently orchestrate, deploy, and manage complex infrastructure setups across different environments with a single, comprehensible set of configuration files.

Question84: What is NixOS?

Answer84:

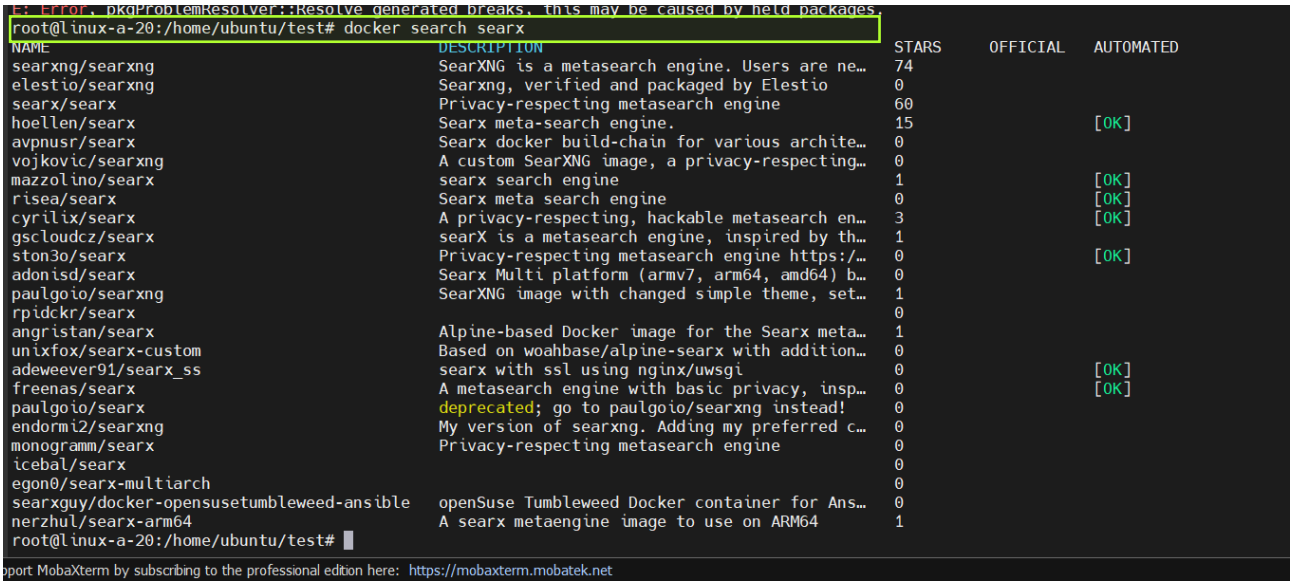
NixOS represents a distinctive Linux distribution that is constructed atop the Nix package manager. One of its notable features is its declarative configuration, facilitating dependable system upgrades through various official channels. Moreover, NixOS boasts specialized tools tailored for DevOps practices and

deployment tasks, streamlining and enhancing the management of system configurations and deployment workflows.

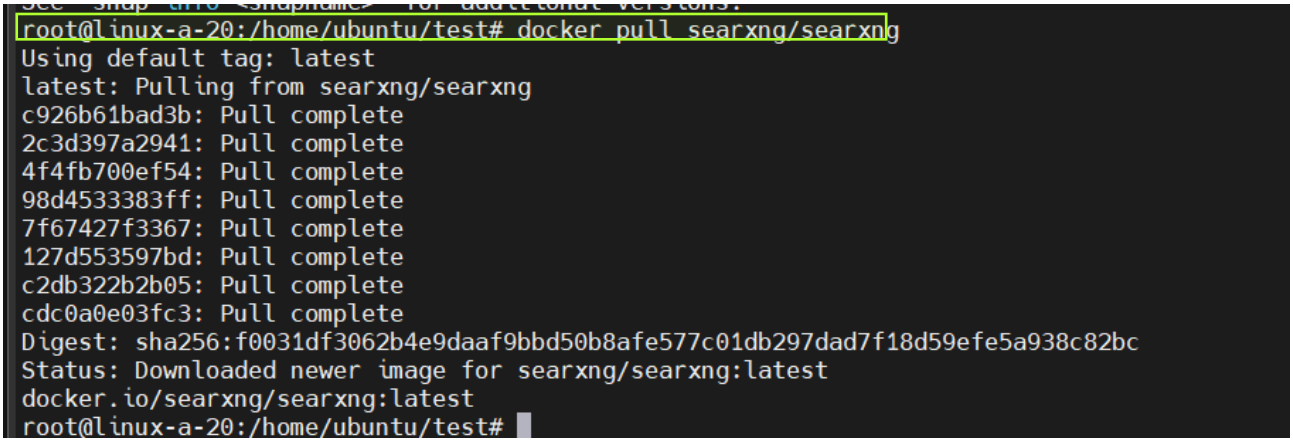
Question85: Install Docker engine (docker.io) package to your server from standard package repository or from the Docker official repository

Answer85:

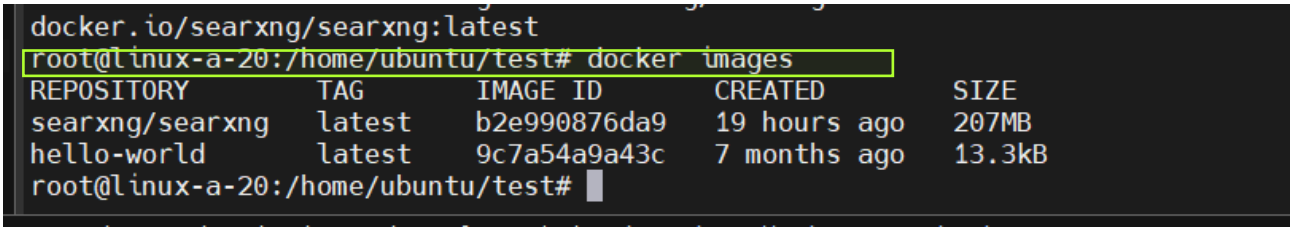
- **Use docker CLI command to search package searx with most stars**



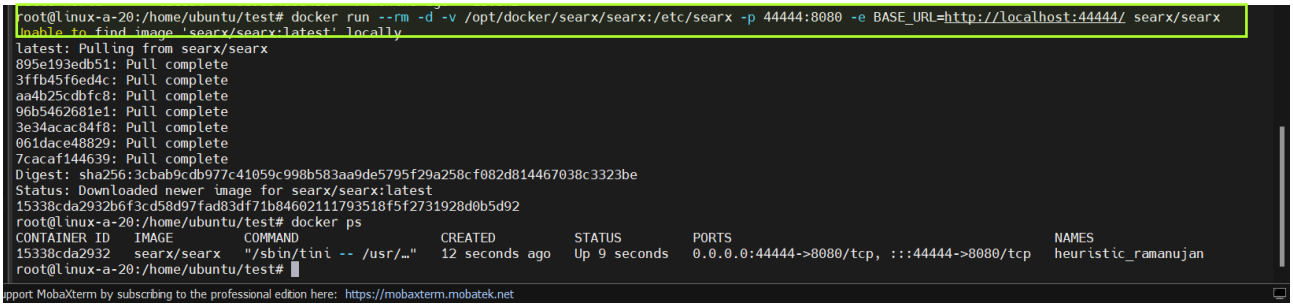
- **Use docker CLI to pull searx/searx Docker container image from Docker repository (Searx is a self-hosted metasearch engine, combining multiple web search engine results)**



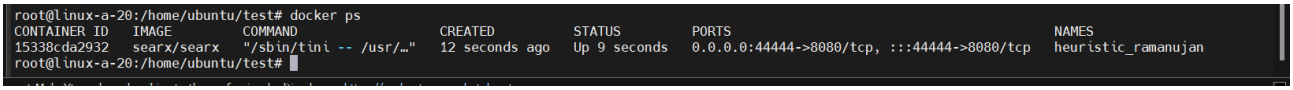
- **List local Docker images with docker images command. What is the image id of Searx?**



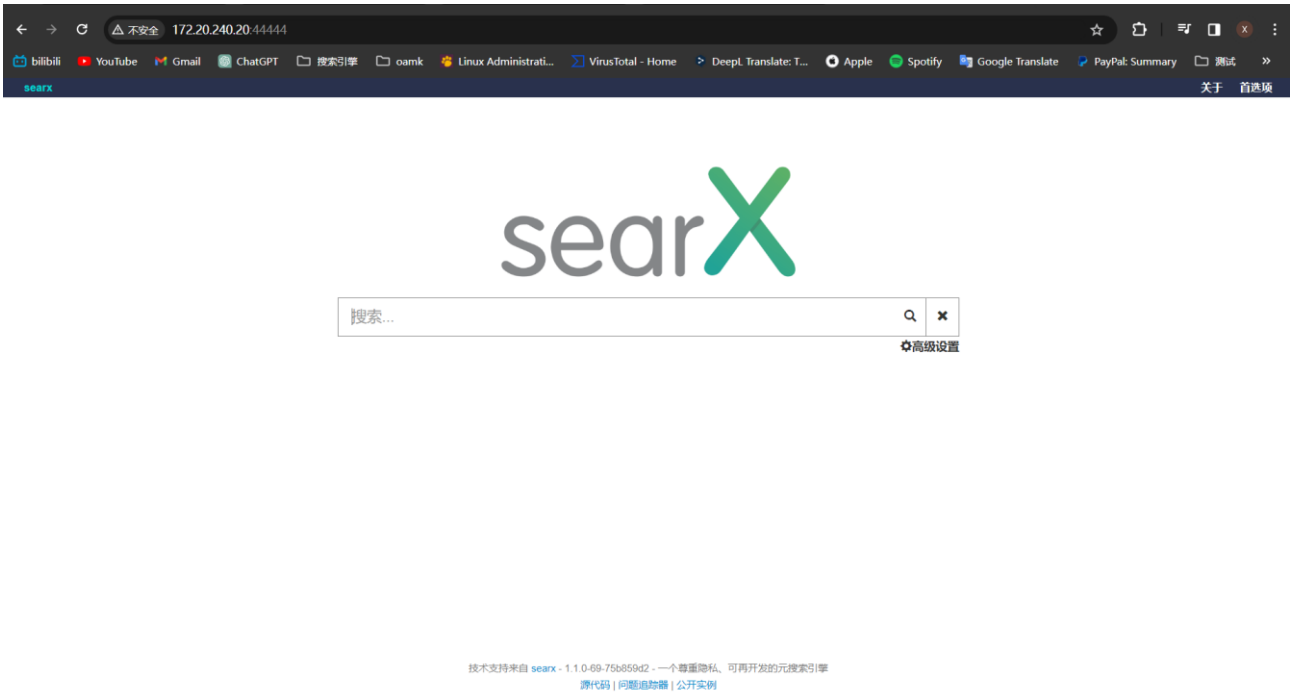
- **Start the Searx container and redirect inbound TCP 8080 port traffic to the Searx container:**



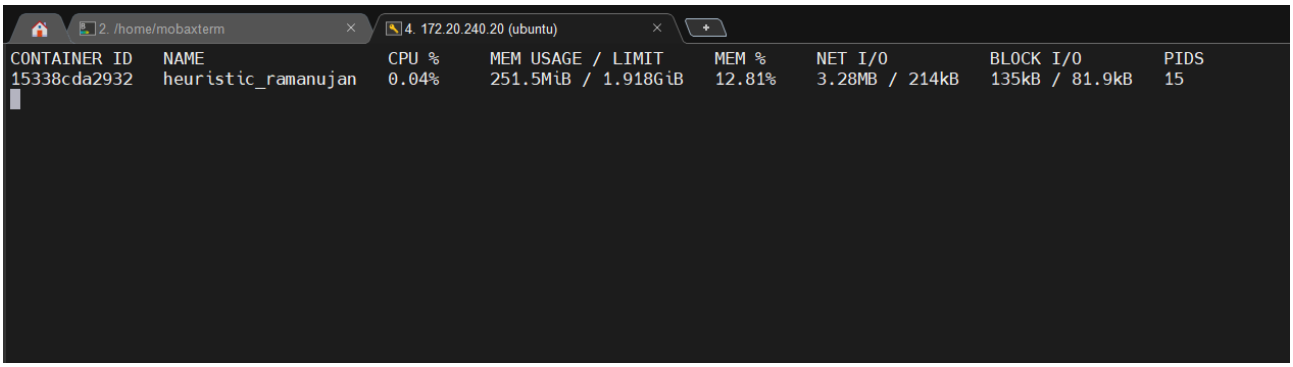
- **Use docker ps to verify that Searx container is running**



- **Verify that you can access your container with your web browser: http://your_server_ip:44444/ (open TCP/44444 port in your host firewall if you are filtering traffic with Netfilter)**



- **Check container stats with interactive docker stats command**



- **Check Searx default settings in /opt/docker/searx/searx/settings.yml file.**
  See https://docs.docker.com/engine/swarm/configs/ and variables like BASE_URL

```
root@linux-a-20:/home/ubuntu/test# cat /opt/docker/searx/searx/settings.yml file
general:
    debug : False # Debug mode, only for development
    instance_name : "searx" # displayed name
    contact_url: False # mailto:contact@example.com
    enable_stats: False # activate /stats page - note: it may leak usage data

brand:
    git_url: https://github.com/searx/searx
    git_branch: master
    issue_url: https://github.com/searx/searx/issues
    docs_url: https://searx.github.io/searx
    public_instances: https://searx.space
    wiki_url: https://github.com/searx/searx/wiki
    twitter_url: https://twitter.com/Searx_engine

search:
    safe_search : 0 # Filter results. 0: None, 1: Moderate, 2: Strict
    autocomplete : "" # Existing autocomplete backends: "dbpedia", "duckduckgo", "google", "startpage", "swisscows", "qwant", "wikipedia" - leave
 it off by default
    default_lang : "" # Default search language - leave blank to detect from browser information or use codes from 'languages.py'
    ban_time_on_fail : 5 # ban time in seconds after engine errors
    max_ban_time_on_fail : 120 # max ban time in seconds after engine errors
    prefer_configured_language: False # increase weight of results in configured language in ranking

server:
    port : 8888
    bind_address : "127.0.0.1" # address to listen on
    secret_key : "546c824a13805a5717633daf5887a3718a1de340d301e90bea58c89676d76376" # change this!
    base_url : http://localhost:44444/ # Set custom base_url. Possible values: False or "https://your.custom.host/location/"
    image_proxy : False # Proxying image results through searx
    http_protocol_version : "1.0"  # 1.0 and 1.1 are supported
upport MobaXterm by subscribing to the professional edition here:  https://mobaxterm.mobatek.net
```

- **Use docker exec to list and explore Searx container filesystem and files. Note: Searx container does not have bash shell but Busybox has SH shell (From CLI: docker exec -t -i your_containerID sh)**

```
root@linux-a-20:/home/ubuntu/test# docker exec -t -i mycontainer /bin/bash
Error response from daemon: No such container: mycontainer
root@linux-a-20:/home/ubuntu/test# docker exec -t -i 15338cda2932 sh
/usr/local/searx #  docker ps
sh: docker: not found
```

- **Stop the container and verify that is has been stopped. Remove the container image if using it anymore**

```
/usr/local/searx # ^C
/usr/local/searx #  docker stop 15338cda2932
sh: docker: not found
/usr/local/searx #
```

Question86: explain shortly what is Robot Framework

Answer86:

Robot Framework is a versatile open-source automation framework utilized for both test automation and robotic process automation (RPA). Supported by the Robot Framework Foundation, this tool is employed by numerous industry-leading companies as a part of their software development process.

Distinguished for its openness and extensibility, Robot Framework seamlessly integrates with various tools, enabling the creation of robust and adaptable automation solutions without incurring any licensing costs.

What sets Robot Framework apart is its user-friendly syntax, employing easily understandable keywords. Additionally, its capabilities can be expanded by integrating libraries implemented in multiple programming languages such as Python, Java, and others. Complemented by a thriving ecosystem, this framework boasts a variety of libraries and tools developed as separate projects, further enriching its functionalities.